Vector Set-Partitioning with Successive Refinement Voronoi Lattice VQ for Embedded Wavelet Image Coding*

Debargha Mukherjee and Sanjit K. Mitra
Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106.
Email: (debu, mitra)@iplab.ece.ucsb.edu

Abstract: *While lattice vector quantization (LVQ) can solve the complexity problem of LBG based vector quantizers, and also yield very general codebooks, a single stage lattice VQ, when applied to high variance vectors result in very large and unwieldy indices, making it unsuitable for applications requiring successive refinement. The goal of this work is to develop a unified framework for progressive uniform quantization of vectors, without having to sacrifice the mean-squared-error advantage of lattice quantization. A successive refinement uniform vector quantization paradigm is developed, where the codebooks in successive stages are all lattice codebooks, each in the shape of the Voronoi region of the lattice at the previous stage. The Voronoi shaped lattice codebook at each stage is called Voronoi lattice VQ (VLVQ). Measures of efficiency of successive refinement are developed. The developed methodology is applied to successively refine vectors of wavelet coefficients in the vector set-partitioning (VSPIHT) framework to obtain an embedded bitstream. The results are compared against the previous Successive Approximation Wavelet Vector Quantization (SA-W-VQ) results of Da Silva, Sampson and Ghanbari for image coding.*

1. Introduction

Considerable interest has been generated in lattice vector quantization (LVQ) over the last couple of decades. Unlike Linde-Buzo-Gray (LBG) based vector quantizers [1] designed from training sequences, structured vector quantizer codebooks obtained as subsets of multidimensional lattices [2]-[5], commonly known as lattice VQ (LVQ), are more appropriate for high resolution quantization because of their generality and the existence of fast encoding and decoding algorithms. Moreover, for uniformly distributed sources, it can be shown that the normalized mean-squared-error per dimension per codepoint [2],[5], for an appropriately chosen multidimensional lattice VQ is significantly lower than that obtained if the individual components of the vector are scalar quantized.

The wavelet transform has recently been shown to be very effective for image compression [6]-[13]. Although plain lattice vector quantization of wavelet coefficient vectors has been successfully employed by several researchers for image compression [7], improved results can be obtained if the benefits of LVQ are combined with the powerful across scale *zerotree* prediction methodology, as in EZW [10] by Shapiro, and SPIHT [11] by Said and Pearlman. While both EZW and SPIHT rely on partial magnitude ordering of wavelet coefficients followed by progressive scalar refinement, and produce embedded bitstreams, SPIHT is more efficient in transmission of significance information to the decoder.

Da Silva et al. [8] developed a vector extension of the EZW algorithm, where a gain-shape type multistage vector quantization method, called Successive Approximation Wavelet Vector Quantization (SA-W-VQ), is used. Points on the first one or two shells of certain multidimensional lattices [2]-[5] are used to describe the shape codebook. Recently Knipe et al. [9] extended the scheme to SPIHT, using modified shape codebooks based on the Λ_{16} lattice, named *ModLVQ*, to yield superior performance. In our earlier work [12],[13], we developed independent of [9], the vector extension of the scalar algorithm of Said and Pearlman, named Vector SPIHT (VSPIHT). Several neighboring coefficients are coded at once using trained, classified, successive refinement VQ [1]. Besides producing good compression results, VSPIHT is more noise resilient, because the balance of bits is shifted to include more quantization bits and less significance bits. While a trained VQ approach to VSPIHT yields good results for a wide range of images and video, it can never be claimed to be truly generic. In this work, we attempt developing a more structured and unified framework for successive refinement uniform vector quantization, where lattice codebooks in the shape of Voronoi regions [2] of multidimensional lattices are used. This scheme is more generic than the gain shape type vector quantization used in [8], [9], where the scale factor can only reduce by a factor between 0.5 and 1. Use of relatively small Voronoi lattice codebooks rather than a single large codebook makes embedding possible, and also eliminates the problem of large indices. The developed scheme is considerably generic, and can be adopted for all applications requiring gradual uniform quantization of vectors. Further, with the recent trend towards multiwavelets, vector based compression is likely to assume greater significance than just in application to quantization of uniwavelet coefficients.

In Section 2, the successive refinement lattice VQ approach is explained in detail. In Section 3, the application of successive refinement lattice VQ to VSPIHT coding of monochrome images using the four dimensional D_4 lattice, is presented. Section 4 concludes the paper.

2. Successive Refinement Lattice VQ

The objective of successive refinement Voronoi lattice VQ is to develop a generic framework for progressive refinement uniform VQ, that can be used for a variety of gradual refinement applications such as embedded coding or progressive transmission.

2.1 Definitions

We first introduce some definitions and notations. A *lattice L* in *n*-dimensional space $\Re^n$, is defined as the set of all integer combinations of a linearly independent set of vectors. That is,

$$L = \{y | y = i_1 a_1 + i_2 a_2 + \ldots i_n a_n, i_k \in Z\} , \quad (1)$$

where $\{a_1, a_2, \ldots, a_n\}$ are a set of *n* linearly independent vectors, and $\{i_1, i_2, \ldots, i_n\}$ are all integers. A *lattice coset or translate* Λ, is obtained from a lattice *L* by adding a fixed translation vector to the points of the lattice. In the rest of the paper, all references to *lattice* will actually denote the general family of lattice cosets, unless otherwise noted.

Around each point *y* in a lattice coset Λ, is its Voronoi region *Vor*(Λ,*y*), consisting of all points in the underlying space which are closer to *y* than to any other lattice point. Formally, it is the closed set given as:

* This work was supported by ONR grant N00014-95-1-1214.

$$Vor(\Lambda, y) = \{x \in \Re^n \,|\, \|x-y\| \le \|x-z\|, y \in \Lambda, \forall z \in \Lambda\} \ . \quad (2)$$

The zero-centered Voronoi region $V_0(\Lambda)$ of a lattice coset Λ, is defined as:

$$V_0(\Lambda) = Vor(\Lambda, y) - y \ , \quad (3)$$

where y is any lattice point in Λ. Note that the zero-centered Voronoi region of all translates of the same lattice is the same.

2.2 Voronoi Lattice VQ (VLVQ)

We now define a new family of lattice VQs, hereafter referred to as *Voronoi Lattice VQ (VLVQ)*. A Voronoi lattice VQ, or VLVQ, is a lattice or lattice coset codebook, in the shape of the zero-centered Voronoi region of a different lattice, usually having a higher scale than the former. The lattice coset from which the codebook is actually constructed is referred to as the *base lattice* of the VLVQ, while the higher scale lattice that determines the shape of the codebook, is referred to as its *shape lattice*. In other words, if Λ_0 be the shape lattice, and Λ_1 be the base lattice of a VLVQ denoted Γ_1 in n-dimensional space, then the intersection of the base lattice Λ_1 with the zero-centered Voronoi region $V_0(\Lambda_0)$ defined by the shape lattice Λ_0, constitutes the VLVQ. The relevance of the shape lattice to a VLVQ is only through the shape of its zero-centered Voronoi region, and is therefore independent of its translations. The base lattice, however, directly influences the VLVQ codebook, and different cosets of the same lattice yield different codebooks.

A desirable property for a VLVQ to possess is to have the zero-centered Voronoi region of the shape lattice completely contained by the base lattice Voronoi regions around the codepoints of the VLVQ. That is:

$$V_0(\Lambda_0) \subseteq \bigcup_{c \in \Gamma} \{Vor(\Lambda_1, c)\} \ . \quad (4)$$

Note that this property is not automatically satisfied by the VLVQ definition above.

To count the number of points in a VLVQ, it is necessary to count the number of points of the base lattice on surfaces of successive scales of the zero-centered Voronoi region of the shape lattice. In general, for every shape-base pair of lattices, a *phi-function* $\Phi(r)$ can be generated to give the number of base lattice points lying on the surface of the zero-centered Voronoi region of the shape lattice scaled by r. Any standard definition of the shape lattice can be considered as being unity scale. In general, these Φ-functions may have to take non-integer arguments r even with suitable definitions of the unity scale shape lattice. A special situation of interest, and often the most useful, is one where the base lattice is a scaled down and possibly translated version of the shape lattice, i.e.

$$\Lambda_1 = \Lambda_0 / r - t \, , \quad (5)$$

t being an arbitrary translation vector. Since this assumes that the unity scale shape lattice with a possible translation, is exactly the base lattice, the function $\Phi(r)$ for a lattice coset now counts the number of points on successive scales of its own zero-centered Voronoi region. This function is so important for a lattice coset, that we call it the *ohm-function* $\Omega(r)$. In Table 1 we present as an example, the Ω-function for the 4-dimensional D_4 lattice. The D_n lattice is defined as the set of all integer n-tuples such that their sum add up to even. In Table 2, we present an alternate Ω-function for a D_4-coset with the translation vector being (1,0,0,0).

Table 1. Ohm function for the D_4 lattice

r	$\Omega(r)$
0	1
2	48
3	96
4	288
5	480
6	912
7	1344
8	2112

Table 2. Ohm-function for the D_4 coset translated by (1,0,0,0).

r	$\Omega(r)$
1	8
2	32
3	120
4	256
5	520
6	864
7	1400
8	2048

2.3 Successive Refinement with VLVQs

The essence of successive refinement lattice VQ is to generate a series of decreasing scale VLVQs, each covering the Voronoi region of the base lattice at the previous higher scale. Given the VLVQ Γ_1 defined in the previous subsection with shape lattice Λ_0 and base lattice Λ_1, we can now construct a lower scale VLVQ, Γ_2, which uses Λ_1 as the shape lattice, and a lower scale lattice coset Λ_2 as the base lattice. Continuing in this manner, a series of decreasing scale VLVQs - Γ_1, Γ_2, Γ_3,..., can be constructed using decreasing scale lattice cosets Λ_0, Λ_1, Λ_2, Λ_3,..., such that the ith VLVQ Γ_i has Λ_i as its base lattice and Λ_{i-1} as its shape lattice.

The above mentioned series of decreasing scale VLVQs provide a convenient framework for successive approximation of vectors. Consider an input vector $x \in \Re^n$, which is to be successively refined. Assume that the distribution of x defines a region R_x in n-dimensional space. It is first quantized by a lattice VQ Γ_0 in the shape of the input distribution, using a lattice Λ_0 as a coarse approximation. Let the quantized vector be denoted u_0. Note that the uncertainty in x has been reduced to the zero-centered Voronoi shaped region of Λ_0 around the chosen codevector u_0. The first VLVQ, Γ_1, next quantizes the approximation error (x-u_0) which lies in the Voronoi region of Λ_0, using the base lattice Λ_1, to obtain a refinement u_1. The uncertainty in x is now reduced to the zero-centered Voronoi region of lattice Λ_1. The second VLVQ Γ_2, then quantizes the error (x-u_0-u_1) to u_2, reducing the uncertainty in x further to the zero-centered Voronoi region of Λ_2. Continuing in this fashion, the final approximation $\hat{x}$ of the vector x is obtained as:

$$\hat{x} = u_0 + u_1 + u_2 + \ldots \quad (6)$$

Thus, the first stage lattice VQ Γ_0, followed by the series of VLVQs Γ_1, Γ_2, Γ_3,..., constitute a successive refinement uniform vector quantization system.

We now state the sufficient conditions for the convergence of the approximation $\hat{x}$ obtained by such a successive refinement system, to an input vector x within the input distribution region R_x. First, the region R_x must be contained entirely by the Voronoi regions of lattice Λ_0 around the codevectors of the first stage LVQ Γ_0. Next, all the VLVQs Γ_1, Γ_2, Γ_3,..., must satisfy the property in Eq. (4). Finally, the non-negative central second moment M_i of the Voronoi region $V_0(\Lambda_i)$ of the successive lattices must converge to 0. These conditions are expressed mathematically as:

$$R_x \subseteq \bigcup_{c \in \Gamma_0} \{Vor(\Lambda_0, c)\} \quad (7a)$$

$$V_0(\Lambda_i) \subseteq \bigcup_{c \in \Gamma_{i+1}} \{Vor(\Lambda_{i+1}, c)\}, \ i \ge 0 \quad (7b)$$

$$M_0, M_1, M_2, \ldots \to 0 \quad (7c)$$

For successive refinement applications we further require that the non-negative sequence in Eq. (7c) is strictly monotone decreasing, i.e. $M_0 > M_1 > M_2 > \ldots$

If indeed, the subset relationships in Eq. (7a) and Eq. (7b) are satisfied with equality for a converging successive refinement system, the most efficient coding scheme is obtained. In this case, absolutely no penalty is incurred in making a transition from single stage uniform quantization to successive refinement quantization. Unfortunately, this rarely happens, and the usual sub optimality introduced by successive refinement is suffered.

Although by definition, there is no restriction on the nature of the lattices Λ_0, Λ_1, Λ_2, Λ_3,..., it makes sense to have them as scaled and possibly translated versions of the same elementary lattice. Indeed, there are only a few lattices for each value of the dimensionality n that yield optimal or near-optimal quantization in terms of the dimensionless mean-squared-error. Further, having scaled versions of the same lattice makes implementation easier. As such, successive lattices are usually constrained such that,

$$\Lambda_i = \frac{\Lambda_{i-1}}{r_i} - t_i, \tag{8}$$

where t_i are arbitrary translation vectors, and r_i are factors by which the uncertainty in the input vector x is reduced at the ith stage. If each r_i is greater than 1, and the conditions in Eq. (7a) and Eq. (7b) are satisfied, convergence of the approximation error is guaranteed.

The principle of successive refinement with VLVQs is demonstrated in Figure 1 by means of the two dimensional hexagonal lattice A_2. Here we see a series of decreasing scale A_2 lattices each covering the hexagonal Voronoi region of the lattice at the previous scale. The lattice at each stage decreases in scale by a factor $r = 4$. Apart from the first stage lattice VQ, all successive stages are Voronoi lattice VQs. Note that in this figure, we have used the zero-centered A_2 lattice as the base lattice in each stage. However, this is not the only option. Figure 2 shows two possible VLVQs for the A_2 shape lattice, one with a quarter-scale translated A_2 base lattice, and the other with a quarter-scale zero-centered A_2 base lattice. Note that both the VLVQs satisfy the condition in Eq. (4). However, the zero-centered one in Figure 2(b) is more efficient, as we see in the next subsection.

We conclude this subsection with a final comment. The codevectors lying on or near the boundaries of a VLVQ does not satisfy the centroid condition for optimality of VQs. Once successive quantization of an input vector has been completed, a better reproduction is obtained if the last stage refinement vector is appropriately corrected if it lies on or near the boundary of the corresponding VLVQ shape. The centroid of the intersection of its Voronoi region with the last stage VLVQ shape, is used as the refinement rather than the codevector itself.

2.4 Index Entropy Coding and Efficiency

When the shape and base lattices in a VLVQ are constrained by Eq. (5), the volume of the Voronoi region of the shape lattice is exactly r^n times the volume of the Voronoi region of the base lattice. As mentioned earlier, the most efficient VLVQ satisfying Eq. (4) to use, would then be the one which has exactly r^n codevectors, with their combined Voronoi regions exactly filling the zero-centered Voronoi region of the shape lattice. Unfortunately, except for the cubic lattice, this condition is seldom satisfied. Let us observe more closely the Voronoi lattice codebooks for the A_2 lattice in Figure 2. Since $r = 4$, and $n = 2$, in this example, the ideal VLVQ should have only $4^2 = 16$ points and still satisfy Eq. (4). Assuming a

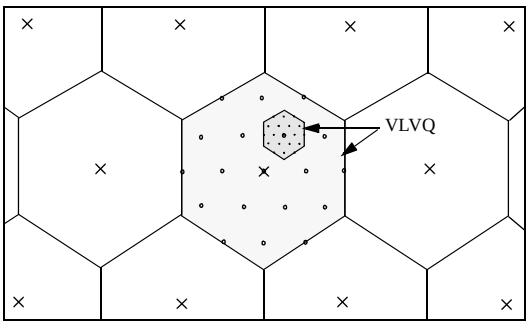

Figure 1. Illustration of multistage Lattice VQ with the A_2 lattice

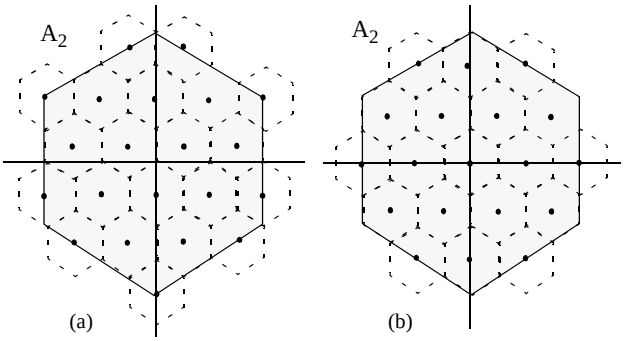

Figure 2. Voronoi Lattice VQs for the A_2 lattice.

uniform distribution of the input vectors within the Voronoi shape, which makes sense in a multistage scenario, the index entropy for this hypothetical VLVQ is exactly 4.0 bits. The real VLVQ in Figure 2(a), however, consists of 21 points. 12 of them have their Voronoi regions entirely included within the zero-centered Voronoi region of the shape lattice, 6 have half of their volumes included, and the remaining 3 have only a third of their volumes included. Under the uniform distribution assumption, the index entropy can be calculated as 4.2866 bits. The VLVQ in Figure 2(b), on the other hand, has 19 points. 13 of them have their entire Voronoi regions included within the zero-centered Voronoi region of the shape lattice, and the remaining 6 have only half their Voronoi regions included. This amounts to an index entropy of 4.1875 bits. As such, the zero-centered base lattice VLVQ in Figure 2(b) is more efficient than the one in Figure 2(a).

In general, for VLVQs constrained as in Eq. (5), the index entropy H under uniform distribution assumptions, is lower bounded by $\log_2 r^n$ bits. That is, $H \geq \log_2 r^n$. The closer the entropy of a practical VLVQ comes to this value, the more efficient it is. A more generic measure of index entropy, independent of the scale factor r and dimensionality n is the normalized index entropy E, defined as:

$$E = \frac{H}{n \cdot \log_2 r} \text{ bits/dimension} \tag{9}$$

E gives a measure of the average number of bits spent per dimension, in reducing the uncertainty region around a vector by an octave, i.e. by a factor of two in each dimension. Note that the lower bound on the normalized index entropy is unity, implying $E \geq 1$. The closer the normalized index entropy of a VLVQ comes to this value, the more efficient it is. Two observations must now be made. First, we conjecture that for any n, and any single lattice based VLVQ, E asymptotically reaches 1 as $r \to \infty$. Second, in n-dimensions, the lattice which yields the optimal quantization in mean-squared-error sense, is not necessarily the most efficient for successive refinement for a given r, in terms of the normalized index entropy. For example, the cubic lattice yields VLVQs with $E = 1$ for any n and any r. However, it is poor in terms of normalized mean-squared-error.

In practice, a Huffman coder or an arithmetic coder can be employed to code the indices close to their entropies. The probability for each codevector is taken as proportional to the volume intersected by the Voronoi region around it, with the zero-centered Voronoi region of the shape lattice.

2.5 Multistage and Tree-Structured Lattice VQs

In a VLVQ based successive refinement lattice VQ scenario, as long as the VLVQ used at a particular stage remain independent of the index chosen at the previous stage, we call the successive refinement scheme *multistage*. If the successive VLVQs are constrained by Eq. (8), the average normalized index entropy for the

VLVQs Γ_1, Γ_2,..., is given by:

$$E_{av} = \frac{1}{n} \cdot \frac{\sum H_i}{\sum \log_2 r_i}, \qquad (10)$$

where H_i is the index entropy for Γ_i, and the summations are over all the VLVQs.

However, the average normalized index entropy can be further reduced by using *tree-structured* codebooks. For example, consider the 6 codevectors on the edge of the Voronoi shape in Figure 2(b). Each of them have only half of their Voronoi regions included within the VQ shape. Therefore, if the codevector chosen at any stage is one of these 6, it will not be necessary to use the full VLVQ in the subsequent stage to code the error. A lattice codebook in the shape of half the Voronoi region should be sufficient to refine the uncertainty in the input vector. Simply using a subset of the next stage full VLVQ whose Voronoi regions contain half the shape Voronoi region, can serve the purpose, and is easily implemented. If the successive codebooks each reduce the scale by the same factor *r*, tree-structuring over multiple stages with an appropriately chosen full VLVQ will only yield a finite number of possible subset codebooks, thereby easing implementation further. A Markov chain modeling can then be used to compute the average index entropy. We now state without proof an important asymptotic result regarding tree-structured codebooks designed using the subset partitioning approach. If an infinite number of stages are used, the terminal codevectors of a tree-structured system designed using VLVQ subsets as above, will cover the entire space evenly. Since the sub optimality due to possible duplications on the boundaries at each stage can now be neglected, the index entropy will reach the entropy of the source. This in turn implies that the lower bound unity of the average normalized entropy for successive refinement is achieved, because there is no advantage in single stage quantization over successive refinement. In practice, because of a finite number of stages, the average normalized entropy remains greater than 1.

2.6 VLVQs based on Voronoi Codes

The index entropy coding approach mentioned above is by no means simple for higher dimensional lattices. The intersecting volumes readily become too complex to compute. More importantly, the use of Huffman coding or Arithmetic coding makes the codes too vulnerable to bit errors. When scaled versions of the same lattice is used for successive refinement, an alternative approach is to use *Voronoi Codes* for the VLVQs without entropy coding of indices. These codes were first proposed by Conway and Sloane [4]. In our notation, Voronoi codes are simply VLVQs satisfying Eq. (5) with integer *r*, and with *t* chosen such that no base lattice codepoints lie on the surface of the zero-centered Voronoi region of the shape lattice. In such cases, the quotient group $\Lambda/r\Lambda$ has order r^n, and therefore, the number of codevectors in the VLVQ is exactly equal to r^n. That is, if *r* is a power of 2, so will be the VLVQ codebook size, and as such, an integral number of bits will be required to specify the codebook index. No entropy coding of the indices are done. Another advantage of using Voronoi codes is the existence of fast encoding and decoding algorithms, as shown by Conway and Sloane [4]. Figure 3 shows an example Voronoi code VLVQ for the A_2 lattice at $r = 4$. The VQ contains exactly $4^2 = 16$ points.

The problem with this approach, however, is that the condition in Eq. (7b) is no longer satisfied. As seen in Figure 3, there are some distinct patches shaded dark, within the Voronoi region of the shape lattice that do not belong to the Voronoi region of any of the codepoints of the base lattice. These regions lead to overload distortion that grow in successive stages. If the ratio of scales *r* in successive stages is sufficiently large, the total volume in the overload region will be small, but cannot be altogether eliminated. One way to alleviate the problem partially is to use scaled up VLVQs at each

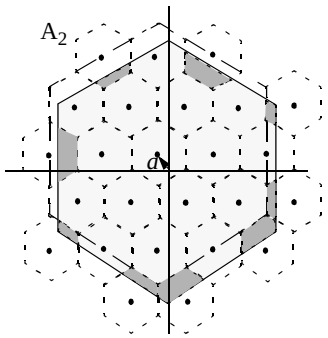

Figure 3. Voronoi Lattice VQ for the A_2 lattice based on Voronoi codes.

stage. If the base lattice used in stage $i-1$ is Λ_{i-1}, the VLVQ designed for stage *i* uses a scaled up shape lattice $(1+\delta)\Lambda_{i-1}$, and a correspondingly scaled up base lattice $\Lambda_i=(1+\delta)\Lambda_{i-1}/r_i - t_i$, instead of the one given by Eq. (8). The factor δ is a small positive number carefully chosen to eliminate the overload region partially or completely. The effective scale factor reduction from one stage to the next is therefore $r_i/(1+\delta)$ instead of r_i.

3. Vector SPIHT Implementation and Results

In vector-based SPIHT for image coding, wavelet transform coefficients in each $H \times V$ window in each band are grouped as a single vector of dimension *HV*. A parent-child relationship between the vectors in different bands is defined as in [11]. The *HV*-dimensional space covered by the wavelet vectors is then partitioned into several classes based on certain pre-defined decision regions that gradually decrease in scale. Specifically, the decision regions are defined by surfaces enclosing the origin that successively decrease in size. Each region is bounded by two surfaces, one on the inside and the other on the outside. The set-partitioning methodology is then used to classify the actual wavelet vectors of the image, in multiple passes, based on the region in *HV*-dimensional space where they lie. Each new pass yields candidate vectors from a new class that lie within the region associated with the pass. The vectors ascertained as significant in a pass are roughly quantized in the same pass, and then progressively refined in successive passes using successive refinement VLVQs.

Based on the above principle, we implemented an image coding system using 4-dimensional wavelet vectors obtained by grouping coefficients in each 2×2 window of each subband. The VLVQs used have D_4 shape lattice and D_4 base lattice. A staggered bit-allocation is used, as we shall shortly see. The wavelet vectors obtained from an image are first scaled so that after scaling, the maximum L_1 norm vector is normalized to a standardized value R_0. This ensures that all the scaled vectors now lie within or on a surface of L_1 norm R_0. The scale factor used is transmitted to the decoder with high precision. The decision regions chosen for partitioning the scaled vectors into classes are shown in the self-explanatory diagram in Figure 4. The first class $Class_0$, is bounded on the outside by a pyramidal surface of L_1 norm R_0, and on the inside by the zero-centered Voronoi region of the D_4 lattice at a particular scale. All other classes are bounded on the outside as well as on the inside by zero-centered D_4 Voronoi regions, with the inside one being at half the scale of the one outside.

Each class defined above is associated with a successive refinement VLVQ system. When a vector is ascertained as significant in a pass, it is coarsely quantized in the same pass with a D_4 lattice VQ, and then progressively refined in each alternate successive pass using VLVQs. The first stage LVQ is in the shape of the region for the class, with the lattice scale being half the scale of the

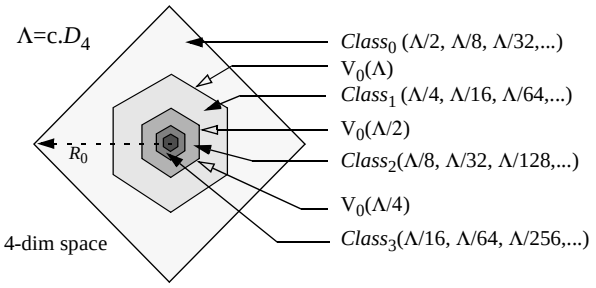

Figure 4. Region Partitioning for Vector SPIHT with Successive refinement VLVQs

D_4 Voronoi region bounding the class on the *inside*. The series of VLVQs used to gradually refine a vector in alternate successive passes, reduce in scale by a factor of 4 at each stage. This staggered refinement allows maintaining high efficiency of successive refinement. In Figure 4, the bracketed sequence beside each class denote the successively lower scale lattices in the successive refinement system, used to quantize vectors in that class. This notation, however, ignores possible translations of the base lattice in the VLVQs.

To design the VLVQs for the D_4 lattice with r = 4 in Eq. (5), two options satisfying Eq. (7a) and Eq. (7b) were investigated. If the base lattice is zero-centered D_4, the number of codevectors in the VQ is 433, obtained by adding the first four rows of Table 1. The index entropy can be computed as 8.55 bits/stage, with the normalized entropy being 1.06875 bits/dimension. If, on the other hand, the base lattice is D_4 translated by (1,0,0,0), the number of codevectors is 416, obtained by adding the first 4 rows in Table 2. The index entropy can be calculated as 8.51 bits/stage, with the normalized entropy being 1.06375 bits/dimension. Therefore, the latter is more optimal.

Two coders based on the above scheme are built. In the first, tree-structured codebooks are used. For each class, the first stage lattice VQ indices are coded using an adaptive arithmetic coder. The refinement VLVQ indices are coded with fixed model arithmetic coders with frequencies proportional to their pre-computed probabilities. Additionally, the significance information is adaptively arithmetic coded as described in [11], using multiple context models. The second coder is designed to be noise resilient, and uses multistage codebooks with fixed length codes for the VQ indices. No arithmetic coding of the indices or the significance information is performed. For both the coders, 5 stages of dyadic decomposition are used. The first two stages use Antonini's 7/9 filters [6], while the remaining three stages use Villasenor's 10/18 filters. Table 3 presents the PSNR results for the two coders at 0.4 bits/pixel for some standard test images, compared against the SA-W-VQ results published in [8] for the D_4 lattice implementation. Even without any form of entropy coding, the PSNR results for the noise resilient coder are superior to SA-W-VQ. Since it uses only about 30-40% of the total bits spent, for the crucial significance information, as opposed to 70-80% for scalar SPIHT, it is very robust to bit errors. Table 4 presents a more comprehensive table of PSNR in dB versus the bit rate for the arithmetic coded VSPIHT coder for several well-known test images.

Table 3. PSNR (dB) results at 0.4 bits/pixel

Gray Image	SA-W-VQ (D_4)	VSPIHT: Tree-structured VLVQ (D_4), entropy coded.	VSPIHT: Multistage VLVQ (D_4), no entropy coding.
Lena	35.17	36.24	35.51
Goldhill	31.01	31.99	31.43
Barbara	29.36	30.23	29.42

Table 4. PSNR (dB) vs. bit rate (bpp) for entropy coded VSPIHT

Rate \ Image	Lena	Goldhill	Barbara	Peppers	Mandrill
0.1	29.61	27.55	23.97	29.29	21.35
0.2	32.89	29.49	26.81	32.32	22.38
0.3	34.45	30.77	28.56	33.44	23.64
0.4	36.24	31.99	30.23	34.86	24.50
0.5	36.92	32.72	31.28	35.35	25.22
0.6	37.60	33.48	32.47	35.86	25.98
0.7	38.18	34.30	34.14	36.34	26.80
0.8	39.26	35.18	34.80	36.94	27.85
0.9	39.64	35.60	35.47	37.65	28.29
1.0	40.02	36.05	36.14	37.93	28.78

4. Conclusions

We have introduced the Voronoi lattice VQ as a means for uniform successive refinement of vectors while preserving the mean-squared-error advantage of lattice quantization. Progressive refinement based on the D_4 lattice, have been applied to wavelet image codng. Investigations on labelling lattice points in a VLVQ, generating ohm-functions for various lattices, and image coding using multiwavelets are in progress.

5. References

[1] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Boston, 1992.

[2] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Information Theory*, vol. IT-28, no. 2, pp. 211-226, March 1982.

[3] J. H. Conway and N. J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Information Theory*, vol. IT-28, no. 2, pp. 227-232, March 1982.

[4] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Information Theory*, vol. IT-29, no. 6, pp. 820-824, Nov. 1983.

[5] H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*, Second edition, Springer-Verlag, New York, 1993.

[6] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing*, vol. 1, pp. 205-220, April 1992.

[7] M. Barlaud, P. Sole, T. Gaidon, M. Antonini, and P. Mathieu, "Pyramidal lattice vector quantization for multiscale image coding," *IEEE Trans. Image Processing*, vol. 3, no. 4, pp. 367-81, July 1994.

[8] E. A. B. da Silva, D. G. Sampson, M. Ghanbari, "A successive approximation vector quantizer for wavelet transform image coding," *IEEE Trans. Image Processing*, vol. 5, no. 2, pp. 299-310, Feb. 1996.

[9] J. Knipe, X. Li, and B. Han, "An improved lattice vector quantization scheme for wavelet compression," *IEEE Trans. Signal Processing*, vol. 46, no. 1, pp. 239-43, Jan 1998.

[10] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3445-62, Dec. 1993.

[11] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243-50, June 1996.

[12] D. Mukherjee and S. K. Mitra, "Vector set partitioning with classified successive refinement VQ for embedded wavelet image coding," *Proc. International Symposium on Circuits and System*, Monterey, California, vol. 4, pp. 25-28, June 1998.

[13] D. Mukherjee and S. K. Mitra, "Vector set partitioning with classified successive refinement VQ for embedded wavelet image and video coding," *Proc. International Conference on Acoustics, Speech, and Signal Process-*

ing, Seattle, Washington, vol. 5, pp. 2809-12, May 1998.