Block-Adaptive Lattice Vector Quantization in Image Coding

Wisarn Patchoo[1] and **Thomas R. Fischer**[2], Non-members

ABSTRACT

A subband image coding algorithm is described based on lattice-based spherical VQ and lattice-based pyramid VQ. The algorithm partitions a subband (or wavelet) decomposed image into blocks of various sizes, depending on their energy and complexity constraints on the enumeration encoding of lattice codevectors. Each block is lattice vector quantized and encoded using a product code. The algorithm is simple and effective, exploiting energy clustering in a wavelet transformed image. Using the integer lattice, the algorithm provides performance slightly better than the Set-Partitioning Embedded Block (SPECK) algorithm, and is competitive with JPEG2000 and the set partitioning in hierarchical trees (SPIHT) algorithm.

Keywords: Lattice Vector Quantization, Image Coding, and Wavelet Transform

1. INTRODUCTION

Subband image decomposition [1], typically using a wavelet transform [2], is an effective method of image representation, supporting several desirable image compression properties, such as scalability, region-of-interest coding, and embedded coding for progressive transmission [3]-[6]. An octave subband structure is commonly used (e.g., [5]-[6]). In all except the lowest frequency subband, the subband (or, referred to interchangeably in this paper, wavelet) coefficients have marginal density that is well-modeled as Laplacian, or generalized Gaussian (GG)with small shape parameter (α), i.e., $\alpha = 0.7$, as demonstrated in Fig. 1 [2], [7]. The subband coefficients also have small correlation [7], but they are generally not memoryless, instead displaying an evident nonlinear energy dependence. This is shown in Fig. 2, comparing the empirical block energy density for contiguous blocks of subband coefficients, to the energy density of blocks of memoryless Laplacian or generalized Gaussian data of the same mean and variance. The empirical block energy density is highly peaked at small energy, and heavy-tailed, reflective of the

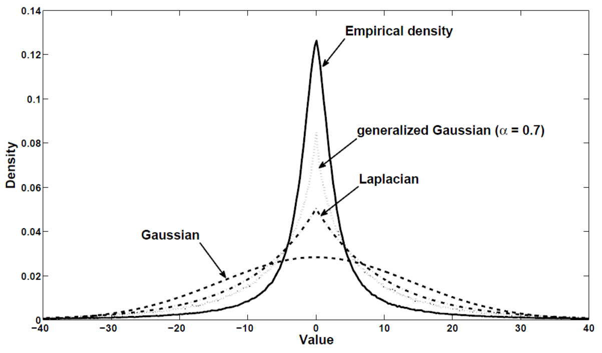

Fig.1: *The marginal density of subband coefficients compared with Gaussian, Laplacian, and generalized Gaussian distribution with* $\alpha = 0.7$ *with the same mean and variance.*

large number of small amplitude subband coefficients, with the larger magnitude coefficients typically clustered, rather than distributed uniformly throughout the subbands. Fig. 2 uses block energy defined as $||\mathbf{x}||_2^2 = \sum_{x_i \in \text{block}} |x_i|^2$, but similar results are obtained defining the block energy as the ℓ_1 norm. Efficient subband image coding algorithms take advantage of such energy dependence, such as by using a quad-tree data structure across [3]-[4] or within subbands [8], or by using local contexts in bit-plane coding [5].

The image subband block energy density can be modeled as a mixture of memoryless Gaussian or Laplacian random variables [9]-[10]. A suitable block-based lattice vector quantizer (LVQ) can be formed using concentric spherical (motivated by the Gaussian mixture model), or pyramidal LVQ (motivated by the Laplacian mixture model). A spherical LVQ using the RE_8 lattice [11] is used in the AMR-WB+ audio coding standard [12]. A pyramid-based LVQ has been used for image coding [13]-[14]. Lattice-based vector quantization offers the potential of relatively simple quantization and granular coding gain [15]. However, the mapping from lattice codevectors to binary codewords can be cumbersome, with complexity that typically grows with the lattice dimension.

In this paper, we propose an image coding algorithm based on lattice-based spherical vector quantization (LSVQ) or lattice-based pyramid VQ (LPVQ), motivated by Fig. 2. Quadtree partitioning is used to divide regions of coefficients into blocks to be lattice vector quantized and encoded. The block sizes are

Manuscript received on October 1, 2012 ; revised on March 14, 2013.

[1] The author is with School of Engineering, Bangkok University, Thailand. 12120, E-mail: wisarn.p@bu.ac.th

[2] The author is with School of Electrical Engineering and Com- puter Science, Washington University, Pullman, USA. 99163,, E-mail: fischer@eecs.wsu.edu

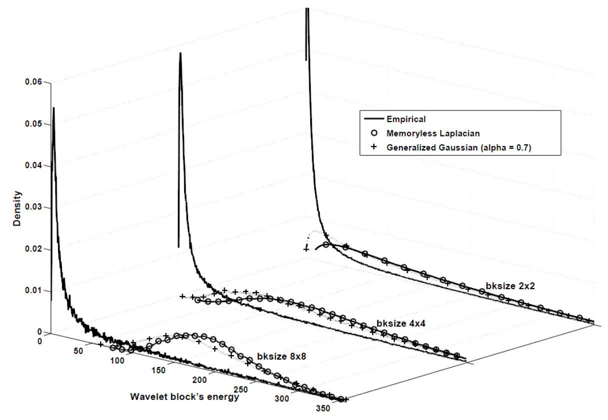

Fig.2: *Empirical density of wavelet block's energy compared to that of memoryless Laplacian and generalized Gaussian vectors with the same mean and variance.*

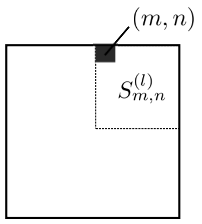

Fig.3: *Set* $S^{(l)}_{m,n}$

selected based on block energy constraints. If a block has too large energy, it is partitioned into smaller blocks. The maximum block energy is selected subject to complexity constraints on the enumeration of lattice codevectors as binary codewords. Blocks of coefficients are vector quantized and encoded using a product code based on LSVQ or LPVQ. The block partitioning is motivated by SPIHT-type algorithms, [8], [16], however, instead of partitioning based on significant and insignificant bit-plane coefficients, the proposed method partitions based on block energy. The resulting bitstream is not embedded, but offers the potential granular gain of the lattice.

2. BLOCK-ADAPTIVE LATTICE VQ

This section briefly describes the basic concepts of the proposed block-based image coding using set partitioning, and introduces the notation and terminology used later in the paper. An image is decomposed into subbands, typically using a wavelet transform. This transformed image, or portions of it, are structured as sets of coefficients. A set is called *e-limited* if its energy is less than a threshold; otherwise it is called *non-e-limited.* The threshold is generally dependent on the number of coefficients in the set (the vector dimension). The block partitioning algorithm begins with large sets of coefficients. Sets are tested, and if non-e-limited, are split into subsets for further testing. Sets that are e-limited are quantized and encoded without further partitioning of the set. This can be done directly, or in a progressive manner. Partitioning can be done based on spatial orientation trees [8], [17] or quadtrees [16]. The goal of partitioning is to keep splitting off clusters of large energy coefficients while maintaining a large set of relatively small energy coefficients which are jointly vector quantized and encoded. By doing this, the energy clustering evident in Fig. 2 can be exploited and the quantization and encoding can be performed efficiently. In this work, partitioning based on quadtrees, similar to [16], is used and the testing condition uses either the ℓ_2 or ℓ_1 norm for set energy. More specifically, an e-limited set in this work implies a set that can be quantized and encoded using a product code based on LSVQ or LPVQ and hence, the vector of set coefficients lies within a bounding sphere or pyramid of respective ℓ_2 or ℓ_1 radius.

Let $c_{i,j}$ and $\hat{c}_{i,j}$ be the unquantized and corresponding quantized version of the wavelet coefficient at coordinate (i,j) of a transformed image of size $N \times N$. Denote a subset (or block) at level l due to quadtree paritidoning, and its corresponding quantized version, as $S^{(l)}_{m,n}$ and $\hat{S}^{(l)}_{m,n}$, respectively, where (m,n) is the coordinate of a given coefficient in $S^{(l)}_{m,n}$ used to refer to that subset, e.g., the coordinate of the coefficient at the block top-left corner, as shown in Fig. 3. Denote subsets (or subblocks) resulting from quadtree partitioning $S^{(l)}_{m,n}$ as $O(S^{(l)}_{m,n})$. Define $E(\hat{S}^{(l)}_{m,n},\nu) = \sum_{\hat{c}\in\hat{S}^{(l)}_{m,n}} |\hat{c}_{i,j}|^\nu$, where $\nu \in \{1,2\}$, as the energy in a block of quantized coefficients. A set $\hat{S}^{(l)}_{m,n}$ is said to be e-limited if

$$E(\hat{S}^{(l)}_{m,n},\nu) \leq T^{(l)} \qquad (1)$$

where $T^{(l)}$ is a predefined threshold for partition level l, and the threshold can vary with partition level.

The LSVQ and LPVQ use as codevectors all lattice vectors within a bounding sphere or pyramid as shown in Fig. 4 and Fig. 5, respectively. Let n denote the vector dimension and Λ_n the lattice. We assume the lattice is of the form described in [19], with $\Lambda_n \subseteq Z^n$, where Z^n is the cubic lattice of dimension n. Let $N_{\Lambda,1}(n,k)$ denote the number of lattice points satisfying $||\mathbf{y}||_1 = k$, and let $N_{\Lambda,2}(n,k)$ denote the number of lattice points satisfying $||\mathbf{y}||_2^2 = k$, $k = 0,1,\ldots$. These values are summarized as the lattice nu-function [13] and theta function [15], respectively. A variable-length code, C_ν, (e.g., a Huffman code) is used to encode the ν-norm energy, with codeword denoted by $\mathbf{c}_\nu(||\mathbf{y}||_\nu^\nu)$. An enumeration code, C_c, uses $\lceil \log_2(N_{\Lambda,\nu}(n,||\mathbf{y}||_\nu^\nu))\rceil$ bits/vector to encode the codevector, conditioned on the energy. Thus, the encoding bits for the quantized block $\hat{S}^{(l)}_{m,n}$ are composed of two parts: 1) a codeword used to specify the sphere or the pyramid, $\mathbf{c}_\nu(||\mathbf{y}||_\nu^\nu)$, and 2) a codeword

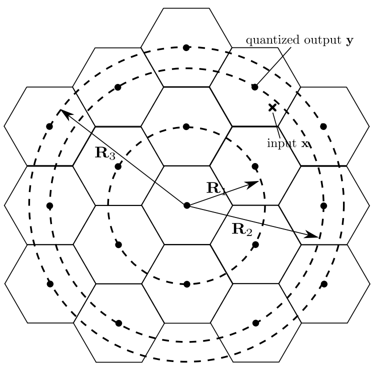

Fig.4: *Lattice Spherical Vector Quantization (LSVQ)*

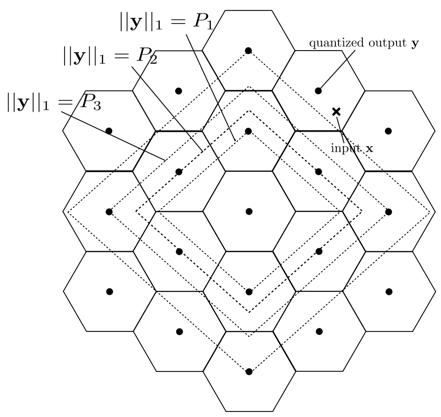

Fig.5: *Lattice Pyramid Vector Quantization (LPVQ)*

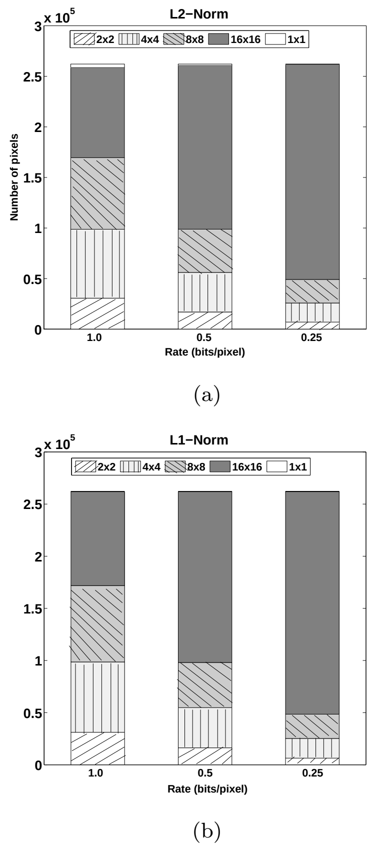

Fig.6: *Number of pixels for each block size: (a) based on ℓ_2-norm; (b) based on ℓ_1-norm:*

used to specify the codevector on a given sphere or pyramid, $\mathbf{c}_c$. Encoding and decoding using codes C_ν and C_c can be implemented primarily using look-up tables.

Fig. 6(a) shows the relative distribution of blocks of various sizes resulting from the quadtree partitioning based on the proposed algorithm and the ℓ_2 energy measure. Similar distributions are obtained for the ℓ_1 case as shown in Fig. 6(b). At low overall encoding rate there are many large blocks and very few singleton blocks, most of the latter from the low frequency subband. Even at large encoding rate, there are relatively few singleton blocks, and the abundance of 2×2 and 4×4 blocks suggest the granular gains of the D_4, E_8, 16-dimensional Barnes-Wall lattice (Λ_{16}), or other lattices might improve the encoding signal-to-noise ratio (SNR).

3. CODING ALGORITHM

3.1 Algorithm

The main idea of the partitioning algorithm is simple. Large subsets are tested according to an energy threshold to determine whether partitioning is required. As soon as subsets have energy within the testing threshold, they are encoded using a product code based on LSVQ ($\nu = 2$) or LPVQ ($\nu = 1$) and are removed from further consideration. Since the order in which subsets are tested is important and needs to be maintained, a list is used to store the testing order. Denote the list of sets as LS. Also, define the binary testing function as

$$EF(\hat{S}^{(l)}_{m,n}, \nu) = \begin{cases} 0, & \text{if } E(\hat{S}^{(l)}_{m,n}, \nu) \le T^{(l)} \\ 1, & \text{if } E(\hat{S}^{(l)}_{m,n}, \nu) > T^{(l)}. \end{cases} \quad (2)$$

Assume N is power of 2. The proposed encoding algorithm is presented in Algorithm 1.

The algorithm uses the functions $LVQ()$ to perform lattice VQ, *encode*() to generate the binary codeword representing a lattice codevector, and $encode_{scalar}()$ to generate the binary codeword for a singleton block. To quantize the block $S^{(l)}_{m,n}$ using LVQ, the vector is formed based on raster scanning. The decoding algorithm is similar to the encoding algorithm, replacing *output* to *input*, *encode()* to *decode()*, and quantization is no longer required. In the algorithm main loop (while loop), the block size is decreased if the block is non-e-limited. The decoder tracks the block size from the sequence of energy-testing bits. The algorithm makes no assumption on the lattice (or lattices) used. However, if the integer lattice is used, lattice quantization can be done

Algorithm 1 Block-Adaptive LVQ

```
Given: subband image $\{c_{i,j}\}$ and chosen $\nu$-norm
Initialize: $LS = \{(0,0)\}$, $l = 0$
Start:
1:  while $LS \neq \phi$ do
2:    for all $(m,n) \in LS$ except those added in current $l$ do
3:      if $N > 1$ then
4:        • $\hat{S}^{(l)}_{m,n} = LVQ(S^{(l)}_{m,n}, N)$;
5:        • output $EF(\hat{S}^{(l)}_{m,n}, \nu)$;
6:        if $(EF(\hat{S}^{(l)}_{m,n}, \nu) = 1)$ then
7:          • add $O(S^{(l)}_{m,n})$ to the end of $LS$;
8:          • remove $(m,n)$ from $LS$;
9:        else
10:         • output $encode(\hat{S}^{(l)}_{m,n}, N, \nu)$;
11:         • remove $(m,n)$ from $LS$;
12:       end if
13:     else
14:       • $\hat{S}^{(l)}_{m,n} = round(S^{(l)}_{m,n})$;
15:       • output $encode_{scalar}(\hat{S}^{(l)}_{m,n})$;
16:       • remove $(m,n)$ from $LS$;
17:     end if
18:   end for
19:   • $N \leftarrow N \gg 2$;
20:   • $l \leftarrow l + 1$;
21: end while
```

in advance, before the algorithm execution, and the partitioning can be done directly to quantized blocks, $\hat{S}^{(l)}_{m,n}$.

3.2 Complexity consideration

In addition to the complexity of the wavelet transform, which also used in the JPEG2000 and SPIHT algorithms, the block-adaptive LVQ algorithm complexity has three main parts: 1) lattice VQ and encoding, 2) subblock energy testing and encoding, and 3) memory requirement. The lattice VQ is composed of three steps: vector quantization, the ν-norm energy encoding, and the enumeration encoding. Vector quantization is very simple when the Z_n lattice is used (just integer rounding operation), however, it can be complicated when other lattices are used. We give further explanation of this in Section 4. The ν-norm energy encoding can be done easily using a look-up table. The enumeration encoding can be cumbersome if the lattice dimension (equivalent to size of image subblock) becomes large, since the number of lattice points needed to be enumerated increases exponentially with the dimension or rate. However, since the allowable subblock sizes are predefined, and the block energy is limited, the encoding can be simply performed using looking-up tables. It should be remarked that in the proposed algorithm the lattice VQ is performed at every partitioning level so that even a simple look-up table operation may require some running time. However, when the Z_n lattice is used, the lattice VQ need only be performed once before the partitioning is done. This helps reducing encoding time. Since the SPIHT algorithm can be considered as implicitly using entropy coded scalar quantization, the lattice VQ in the proposed algorithm has additional complexity unless the Z_n lattice is used. However, for certain lattices, such as the D_n and E_8 lattices, this additional complexity is very modest [15].

For subblock energy testing, during the partitioning each subblock energy is tested against predefined energy thresholds. Since a large subblock is composed of a group of smaller subblocks (corresponding to the quadtree partitioning), and the a large block's energy is the sum of its subblock energies, a linked list data structure can be used and the subblock energy calculation performed only once before coding the encoding algorithm is executed. During the coding, a specified subblock's energy can be obtained from that list.

Finally, the algorithm's memory requirements are similar to the SPIHT algorithm. In addition to the wavelet transformed image, the partitioning requires the implementation of a linked list, as well as the list of subblock energies. Also, the proposed algorithm requires memory for storing the enumeration look-up tables, which may be large if large allowable block sizes or energy thresholds are used.

4. SIMULATION RESULTS

To evaluate the performance of the proposed algorithm, a five-level subband decomposition is generated using a 9/7 biorthogonal filter bank [4], [5]. At first, the simulation uses integer (cubic) lattice VQ for simplicity. Later on, the simulation using other lattices will be investigated. Both LVQ based on ℓ_1 (LPVQ) and ℓ_2 norms (LSVQ) are used. A Huffman code is generated for C_ν using empirical energy probabilities for each partition level. The largest allowed block size is 16×16 ($N = 16$) so that the allowed possible block sizes due to partitioning are 16×16, 8×8, 4×4, 2×2, and 1×1. Raw energy-testing bits are entropy coded (using arithmetic coding) to exploit the statistical dependence among them. Note that the 9/7 biorthogonal filter is used in the simulation since it is commonly used in most image coding algorithm literatures so that the performance comparison with other image compression algorithms can be done directly. Moreover, by the study using impulse and step response criteria, and using Holder regularity in [18] shows that at the same compression ratio, the 9/7 biorthogonal filter bank offers highest PSNR among all of the minimum order biorthogonal filter banks with combined analysis/synthesis filter lengths of ≤ 36. In general, any other perfect reconstruction filter bank can be used together with the proposed

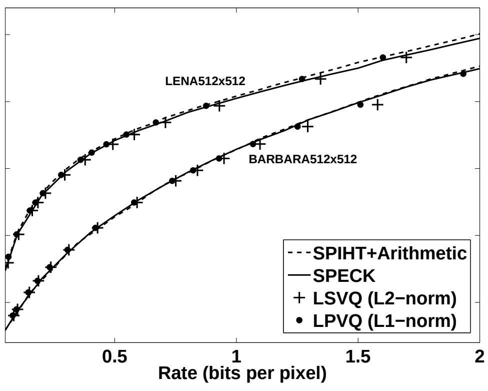

Fig.7: *Performance comparison of LSVQ (ℓ_2) and LPVQ (ℓ_1) with SPIHT and SPECK for "Lena" and "Barbara" images*

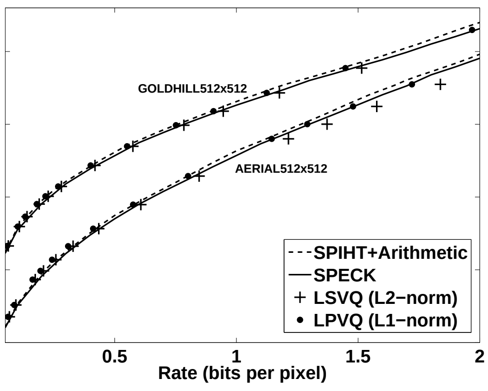

Fig.8: *Performance comparison of LSVQ (ℓ_2) and LPVQ (ℓ_1) with SPIHT and SPECK for "Goldhill" and "Aerial" images*

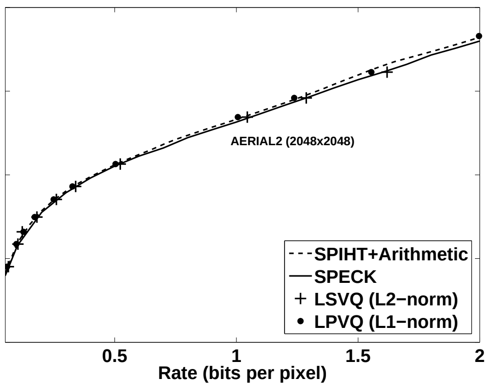

Fig.9: *Performance comparison of LSVQ (ℓ_2) and LPVQ (ℓ_1) with SPIHT and SPECK for "Aerial2* (2048×2048)*" images*

algorithm.

The simulation results are shown in Fig. 7-9 compared with the SPIHT and the SPECK algorithm (SPECK is done using QccPack [26]). The performance is slightly superior to SPECK, and is competitive with the SPIHT algorithm (using adaptive arithmetic coding), with LPVQ performing a little better at low and moderate encoding rates. The LPVQ performs slightly better than the LSVQ. Note that the total encoding rate would increase about 3-5% without entropy coding of the energy-testing bits. So, there is a little gain by entropy coding those bits. About 5-10% of the total encoding rate is used to encode the energy-testing bits.

Figure 6 suggests the possibility that better covering lattices (higher granular gain compared with the Z lattice) might be used, based on the partitioned block sizes, to possibly allow improvement in LVQ SNR due to granular fidelity. Since there are four possible allowed block of sizes, 16×16, 8×8, 4×4, and 2×2, the vectors formed by these blocks are of dimension 256, 64, 16, and 4, respectively. Then, we might be able to use better lattices with dimensions corresponding to those vector dimensions. However, for vectors of size 256 or 64, using other better lattices rather than integer lattice may require very large computatial complexity. To see this, let us briefly describe the notion of the coset codes introduced by Forney [19]-[20]. Forney shows that a family of Barnes-Wall lattices and its principal sublattices can be generated by iteratvely applying a simple construction called the *squaring construction.* He also points out the close relationship between this family of lattices and the family of Reed-Muller codes (RM) [21]. More specifically, denote the lattice generated by the iterated squaring construction as $\Lambda(r,n)$, where r is an integer and the lattice dimension is $N = 2^{n+1}$. $\Lambda(r,n) = Z^N$ for $r \geq n$, and $\Lambda(r,n) = R^{-r}\Lambda(0,n)$ for $r \leq 0$, where R is the *rotation operator.* The Barnes-Wall lattices are equivalent to $\Lambda(r,n)$ when $r \geq 0$. Forney shows that $\Lambda(r,n)$ can be constructed as the union of the $2^{(m/2)}Z^{2N}$ lattice translated by codewords of Reed-Muller codes, where $m = n-r$ when $n-r$ is even, and $m = n-r+1$ when $n-r$ is odd. For example, using Forney's expression, the Gosset (E_8) lattice can be written as

$$E_8 = \Lambda(0,2) = 2Z^8 + (8,4,4),$$

where the sum is a direct sum and $(8,4,4)$ is the (n,k,d) Reed-Muller code with block length n, k information bits, and minimum Hamming distance d [21]. The above expression shows that the E_8 lattice is obtained as the union of Z^8 lattice scaling by a factor 2 and translated by each codeword of the $(8,4,4)$ RM code. It also tells us that performing LVQ using the E_8 lattice is equivalent to performing LVQ using the $2Z^8$ lattice plus its 15 cosets (the $2Z^8$ lattice is itself the zero coset).

Now, consider the following choice of lattices : 256-dimensional Barnes-Wall lattice, Λ_{256}, for vectors of size 256, and 64-dimensional Barnes-Wall lattice, Λ_{64} for vector dimension 64 [15]. Using Forney's expression, Λ_{64} can be represented as

$$\begin{aligned}\Lambda_{64} &= \Lambda(0,5)\\ &= 8Z^{64} + (64,7,82) + 2(64,42,8)\\ &\quad + 4(64,63,2), \qquad (3)\end{aligned}$$

where $(64,7,82), (64,42,8),$ and, $(64,63,2)$ are Reed-Muller codea and the sum is a direct sum. Eq. (3) suggests that LVQ using Λ_{64} is equivalent to LVQ using the $8Z^{64}$ (the 64-dimensional integer lattice scaled by the factor 8) combined with additional similar vector quantizations using its $2^7 + 2^{42} + 2^{63}$ cosets. For Λ_{256}, the number of cosets is far larger than this. Due to the complexity of using the 256- or 64-dimensional Barnes-Wall lattices for VQ, we instead use the simple integer lattice for these dimensions. The 16-dimensional Barnes-Wall lattice, Λ_{16} and D_4 lattice [15] are used to quantize 16-dimensional and 4-dimensional vectors, respectively. An effective quantization algorithm for the D_n lattice is described in [15]. The Λ_{16} lattice can be expressed as [20]

$$\Lambda_{16} = 2D_{16} + (16,5,8). \qquad (4)$$

The above expression suggests a simple way to perform LVQ using Λ_{16}. Let $\mathbf{c}_i$, for $i = 1,\dots,32$, be the binary codewords of the (16,5,8) RM code. Let's denote the 16-dimensional input vector formed by an image block of size 4×4 as $\mathbf{x}$ and the lattice codevector as $\mathbf{y}$. Then, LVQ using Λ_{16} can simply be done as follows.

0) Let $\mathbf{y} = VQ_{2D_{16}}(\mathbf{x})$ denote lattice VQ using the $2D_{16}$ lattice.
1) For $i = 1$ to 32, form $\mathbf{y}_i = \mathbf{c}_i + VQ_{2D_{16}}(\mathbf{x} - \mathbf{c}_i)$.
2) Find $i^* = \text{argmin}_i ||\mathbf{x} - \mathbf{y}_i||^2$.
3) The Λ_{16} codevector closest to $\mathbf{x}$ is $\mathbf{y}_{i^*}$.

Since the proposed algorithm with LPVQ outperformed LSVQ in Section 4, the subsequent simulation results are based on LPVQ. Table 1 shows the simulation results of the proposed algorithm using only integer lattice, and that using the integer lattice together with the D_4 and Λ_{16} lattices for 4-dimensional and 16-dimensional vectors, respectively. To compare with the results when only the integer lattice is used, the D_4 and Λ_{16} lattices are scaled by the factors 1/1.1892 and 1/1.6818, respectively, to have the same point density as the integer lattice[1][2].

[1] D_4 has point density half of that of Z^4. This corresponds to a rate difference of 1 bit per 4 dimensions, or 0.25 bits per dimension. It implies that by scaling D_4 lattice with the factor g, denote scaled D_4 as D_4^g, the rate of D_4^g must be higher than unscaled D_4 by 0.25 bit. Using this and by the fact that at high rate, the encoding rate can be approximately as $0.5\log_2(\sigma^2/D)$, we get $g^2 = 1/(2^{0.5})$, or $g = 1/(2^{0.25}) = 1/1.1892$.

[2] Similar to D_4, point density of Λ_{16} is $1/(2^{12})$ of that of Z^{16}. This corresponds to a rate difference of 12 bits per 16 dimensions, or 0.75 bits per dimension. Then, we get $g^2 = 1/(2^{2(0.75)})$, or $g = 1/(2^{0.75}) = 1/1.6818$.

Table 1: *Performance comparison of proposed method with LPVQ using integer lattice, and D_4 and Λ_{16} lattices*

Image	**Rate (bpp.)**	**PSNR (dB)**	
		LPVQ$_Z$	**LPVQ$_{Lat}$**
Lena	0.125	31.13	30.85
	0.25	34.05	33.75
	0.5	37.15	36.85
	1.0	40.54	40.15
Barbara	0.125	25.29	25.19
	0.25	28.05	27.67
	0.5	31.59	31.34
	1.0	36.53	36.12
Goldhill	0.125	28.48	28.32
	0.25	30.54	30.25
	0.5	33.11	32.89
	1.0	36.68	36.17
Aerial	0.125	23.54	23.34
	0.25	25.86	25.49
	0.5	28.83	28.43
	1.0	33.14	32.57
Aerial2	0.125	26.60	26.47
	0.25	28.54	28.41
	0.5	30.65	30.52
	1.0	33.45	33.36

LPVQ$_Z$: LPVQ using Z-lattice.
LPVQ$_{Lat}$: LPVQ using Z together with D_4 and Λ_{16} lattices.

From the table, it is perhaps surprising that the simulation results using the integer lattice together with D_4 and Λ_{16} lattices (LPVQ$_{Lat}$) provides lower PSNR than using only the integer lattice (LPVQ$_Z$) at the same encoding rate. However, this is consistent with the work of Gao, Belzer, and Villasenor [22], that shows that at low encoding rate, LVQ using the Z-lattice provides performance superior to LVQ using other better covering lattices such as the E_8 or Leech lattices. This follows from the observation that wavelet coefficients can be modeled well using a generalized Gaussian distribution [23] with low shape parameter (also shown in Fig. 1) and the Voronoi regions of the Z-lattice are better suited with the anisotropicity of the generalized Gaussian density which has more vectors along the coordinate axes than off-axis. This effect influences significantly the performance of LVQ at low encoding rates. For high encoding rates, LVQ using lattice with higher granular gain still provides better performance. For our simulations with the Lena image, this occurs at encoding rates higher than about 2 bpp, and LPVQ$_{Lat}$ begins providing better performance over LPVQ$_Z$. It should be emphasized that this particular effect occurs with input source of low-shape-parameter generalized Gaussian distribution, but not of Gaussian distribution.

Table 2: *Performance comparison for several images*

Coding Method	**PSNR (dB)**					
	0.0625 bpp.	**0.125 bpp.**	**0.25 bpp.**	**0.5 bpp.**	**1.0 bpp.**	**2.0 bpp.**
Lena (512 × 512)						
JPEG2000	28.05	31.22	34.28	37.43	40.61	44.84
SPIHT-AC	28.38	31.10	34.11	37.21	40.41	45.07
SPECK	28.16	30.96	33.99	37.09	40.24	44.72
LPVQ$_Z$	28.54	31.13	34.05	37.15	40.54	45.34
Barbara (512 × 512)						
JPEG2000	23.38	25.55	28.55	32.48	37.37	43.17
SPIHT-AC	23.35	24.85	27.58	31.40	36.41	42.65
SPECK	23.36	24.93	27.69	31.48	36.44	42.46
LPVQ$_Z$	23.53	25.29	28.05	31.59	36.53	42.52
Goldhill (512 × 512)						
JPEG2000	26.56	28.33	30.46	33.18	36.58	41.93
SPIHT-AC	26.72	28.37	30.48	33.08	36.56	42.00
SPECK	26.62	28.27	30.33	32.85	36.32	41.57
LPVQ$_Z$	26.91	28.51	30.54	33.11	36.68	41.70
Aerial (512 × 512)						
JPEG2000	21.27	23.10	25.42	28.71	33.19	39.82
SPIHT-AC	21.52	23.20	25.57	28.73	33.17	39.81
SPECK	21.42	23.13	25.44	28.51	32.85	39.53
LPVQ$_Z$	21.92	23.54	25.86	28.83	33.14	39.59
Aerial2 (2048 × 2048)						
JPEG2000	24.63	26.51	28.58	30.64	33.28	38.14
SPIHT-AC	24.63	26.52	28.49	30.60	33.32	38.22
SPECK	24.51	26.40	28.39	30.53	33.16	37.99
LPVQ$_Z$	24.83	26.60	28.54	30.65	33.45	38.29

The LPVQ$_Z$ encoding performance is compared to other well-known coding methods. The comparison is shown in Table 2. The JPEG2000 encoding is done using OpenJPEG [25] and the SPECK encoding is done using QccPack [26]. From Table 2, it can seen that the performance of LPVQ$_Z$ is competitive with JPEG2000 and SPIHT-AC, but is slightly better than SPECK. At low encoding rate ($<=$ 0.5 bpp), LPVQ$_Z$ slightly outperforms the other encoding methods, except JPEG2000. For some images such Goldhill and Aerial and at very low encoding rates, LPVQ$_Z$ provides PSNR increase of about 0.2-0.4 dB. This suggests that the proposed method is particularly effective at low encoding rates.

5. CONCLUSION

In this paper, we propose an algorithm based on LPVQ and LSVQ for image coding. The proposed algorithm partitions the subband (or wavelet) image into blocks of various sizes by comparing ℓ_1 or ℓ_2 block energy, respectively, with the thresholds defined according to complexity constraints on enumeration encoding of lattice points. We show that based on the proposed algorithm the energy clustering can be exploited effectively. From the simulations, the proposed algorithm with LPVQ provides a performance little better than SPECK algorithm, and competitive with JPEG2000 and SPIHT using arithmetic coding. The proposed algorithm works very well at low encoding rates. Coding blocks or sets of coefficients using LVQ also allows the possible improvement in SNR by using better covering lattices, such as D_4 or E_8 or Λ_{16}. However, the above results show that LVQ using better covering lattices to encoding wavelet coefficients gives higher PSNR over LVQ using Z-lattice only at higher encoding rates. It should be noted that for some images the granular fidelity of lattices may possibly provide better visual quality of decoded image compared with SPIHT [27]. Future work will emphasize modifying the LSVQ and LPVQ encoding to provide an embedded bitstream which will allow progressive transmission of images.

References

[1] J. W. Woods, and S. D. O'Neil, "Subband coding of images," *IEEE Trans. on Acoust. Speech and Signal Proc.,* vol. 34, pp. 1278-1288, Oct. 1986.

[2] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing*, vol. 1, no. 2, pp. 205-220, 1992.

[3] J. M. Shapiro, "Embedded image coding using ze-

rotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3445-3462, 1993.

[4] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 243-250, 1996.

[5] *Information technology – JPEG 2000 image coding system: Core coding system*, ISO/IEC 15444-1, 2004.

[6] D. Taubman and M. W. Marcellin, *JPEG2000: Image compression fundamentals, standards and practice*, Springer, 2001.

[7] N. Tanabe and N. Farvardin, "Subband image coding using entropy-coded quantization over noisy channels," *IEEE J. Sel. Areas Commun.*, vol. 10, June 1992.

[8] W. A. Pearlman, A. Islam, N. Nagaraj, and A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 11, pp. 1219-1235, 2004.

[9] J. K. Su and R. M. Merserau, "Coding using Gaussian mixture and generalized Gaussian models," *Conf. Proceedings*, ICIP, pp. 217-220, 1996.

[10] L. Guillemot, Y. Gaudeau, S. Moussaoui, and J. M. Moureaux, "An analytical Gamma mixture based rate-distortion model for lattice vector quantization," *EUSIPCO*, 2006.

[11] M. Xie and J. P. Adoul, "Embedded algebraic vector quantization (EAVQ) with application to wideband audio coding," *Conf. Proceeding*, ICASSP, pp 240-243, 1996.

[12] *Extend AMR Wideband Codec; Transcoding functions*, 3GPP TS 26.290, 2005.

[13] 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. 367-381, 1994.

[14] Z. Mohd-Yusof, T. R. Fischer, "An entropy-coded lattice vector quantizer for transform and subband image coding," *IEEE Trans. Image Processing*, vol.5, no.2, pp.289-298, 1996.

[15] J. H. Conway and N. J. A. Sloane, *Sphere Packing, Lattices, and Groups*, 3rd Ed., New York: Springer-Verlag, 1999.

[16] J. Andrew, "A simple and efficient hierarchical image coder," *Conf. Proceeding*, ICASSP, pp. 658-661, 1997.

[17] C. Chrysafis, A. Said, A. Drukarev, A. Islam, and W. A. Pearlman, "SBHP - a low complexity wavelet coder," *Conf. Proceeding*, ICASSP, pp. 2035-2038, 2000.

[18] J. Villasenor, B. Belzer, and J. Liao, "Wavelet filter evaluation for image compression," *IEEE Trans. Image Processing*, vol. 4, no. 8, pp. 1053-1060, 1995.

[19] G. D. Forney, "Coset codes-I: Introduction and Geometrical Classification," *IEEE Trans. on Inform. Theory*, vol.34, no.5, pp. 1123-1151, 1988.

[20] G. D. Forney, "Coset codes-II: Binary lattices and related codes," *IEEE Trans. on Inform. Theory*, vol.34, no.5, pp. 1152-1187, 1988.

[21] S. Lin and J. D. Costello, *Error Control Coding: Fundamentals and Applications*, 2nd ed., Prentice-Hall, June, 2004.

[22] Z. Gao, B. Belzer, and J. Villasenor, "A comparison of the Z, E_8, and Leech lattices for quantization of low-shape-parameter generalized Gaussian sources," *IEEE Signal Processing Letters*, vol. 2, no. 10, pp. 197-199, Oct 1998.

[23] T. R. Fischer, "A pyramid vector quantizer," IEEE Trans. on Inform. Theory, vol. 32, pp. 568-583, 1986.

[24] W. A. Pearlman, A. Islam, N. Nagaraj, A. Said., "Efficient, lowcomplexity image coding with a set-partitioning embed-ded block coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 11, pp. 1219-1235., Nov. 2004.

[25] OpenJPEG, *http://www.openjpeg.org.*

[26] QccPack, *http://qccpack.sourceforge.net.*

[27] B. A. Banister, T. R. Fischer, "Quadtree classification and TCQ image coding," *IEEE Trans. on Circuits and Systems for Video Technology*, vol.11, no.1, pp. 3-8, Jan 2001.

Wisarn Patchoo received Bachelor degree from Kasetsart University in 2000 and Master degree from King Mongkut's Institute of Technology Ladkrabang in 2003. Both are in Electrical Engineering. From 2002 to 2005, he was a Intelligent Network Engineer at the Huawei Technologies (Thailand). From 2006 until now, he is with School of Engineering, Bangkok University, Thailand. He received Ph.D in Electrical and Computer Engineering from Washington State University in 2011. His research interests are data compression, signal and image processing.

Thomas R. Fischer received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Massachusetts, Amherst, and the Sc.B. degree magna cum laude from Brown University.

From June 1975 until August 1976, he was a Staff Engineer at the Charles Stark Draper Laboratory, Cambridge, MA. From 1979 until 1988 he was with the Department of Electrical Engineering, Texas A&M University. Since January 1989 he has been a Professor in the School of Electrical Engineering and Computer Science at Washington State University. From 1999 until 2004 he was Director of the School of EECS. His current research interests include data compression, image and video coding, joint source/channel coding, digital communications, and digital signal processing.

Professor Fischer has served as Associate Editor for Source Coding for the *IEEE Transactions on Information Theory*, a member of the Information Theory Society Board of Governors, Secretary of the Signal Processing and Communication Electronics Technical Committee of the IEEE Communications Society, and program evaluator for ABET. He is a regular reviewer for several journals, and has served on the Program Committee for several Workshops, Symposia, and Conferences. In 1987 Professor Fischer received an outstanding teaching award from the College of Engineering at Texas A&M University. Five times he has received departmental teaching awards at Washington State University. He was a co-recipient of the IEEE Signal Processing Society's 1993 Senior Award in the Speech Processing Area. In 1996 he was elected Fellow of the IEEE.