IMAGE COMPRESSION WITH A VECTOR SPECK ALGORITHM

Chih-chien Chao and Robert M. Gray

Electrical Engineering Department, Stanford University
350 Serra Mall, Stanford CA 94305
{jichien,rmgray@stanford.edu}

ABSTRACT

SPIHT is an efficient image compression algorithm based on zerotrees. The significant wavelet coefficients are located by a series of set partitioning operations and then scalar quantized. Block-based algorithms inspired by SPIHT such as AGP and SWEET have good performance, but they are not embedded. Pearlman et al. proposed a block-based SPECK algorithm using set partitioning of embedded blocks to exploit the energy clustering characteristics of the coefficients while the bit stream remains embedded. We here propose a variation on SPECK using vector quantization to code the significant coefficients. Different VQ techniques including TSVQ and ECVQ are also considered. Vector SPECK shows a performance improvement over JPEG 2000 at the cost of added complexity.

1. INTRODUCTION

Said and Pearlman [1] proposed the SPIHT algorithm to scan the wavelet coefficients using a set-partitioning method based on the concept of zerotrees. Block-based algorithms have recently become popular because it is easier to implement scalable image compression through blocks. Several block-based algorithms inspired by SPIHT including AGP and SWEET have good performance, but are not embedded. Pearlman et al. proposed a "set-partitioning embedded block" (SPECK) [2] algorithm without using zerotrees to exploit the energy clustering characteristics within the subbands while the output bit stream remains embedded. SPECK can also be used to generate embedded bit streams for code blocks of JPEG 2000. Most SPIHT-related algorithms focus on coding the significance map of the coefficients. The significant coefficients are coded by a simpler scalar quantizer. It is reasonable to consider encoding these significant coefficients as vectors since the adjacent wavelet coefficients are not independent. Mukherjee and Mitra [3] have explored a similar idea in their vector SPIHT algorithm by using 4D vectors to code the significant coefficients. In a similar vein, we here use vector quantization to code the significant coefficients for SPECK.

Partially supported by the National Science Foundation under Grant CCR-0073050.

2. VECTOR FORMING

Wavelet coefficients in different subbands maintain their orientations. The cross correlations between elements of a vector can be higher if the vector is formed in accordance with proper orientation. A VQ codebook trained from a highly correlated source generally has a lower distortion. We use a 9/7 biorthogonal wavelet transform with 5 decomposition levels in our experiment. There are 16 subbands in total (Figure 1). We create 4D vectors with different shapes based on their orientations for coefficients of subband 0 to 9. The original set partitioning procedure method in SPECK[2] is depicted in Figure 2(a). The basic coding unit of SPECK is a pixel. The encoder continues partitioning any significant set until the set contains only one pixel. In vector SPECK, the basic coding unit is a vector. We start with the same sets and continue partitioning any significant set that is larger than a 4×4 block using the same principle until the significant set is a 4×4 block. Further partitioning depends on the orientation. The shapes of vectors within the new sets are either square or rectangular as can be seen in Figure 2 (b – d). The magnitudes of coefficients at higher subbands are generally smaller than those of the lower bands, so the bit requirement is lower. It is reasonable to consider using vectors with higher dimensions to further exploit the correlations among adjacent coefficients. We partition a significant 4×4 block at a higher subband into two 8D vectors as in Figure 3.

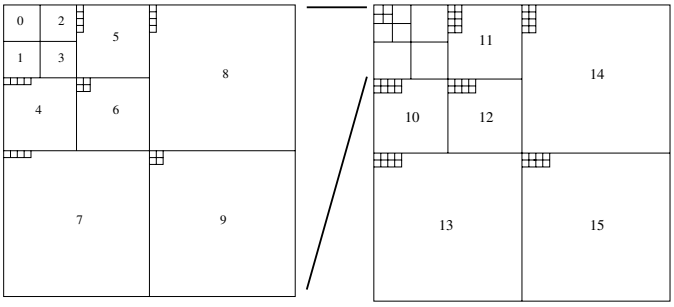
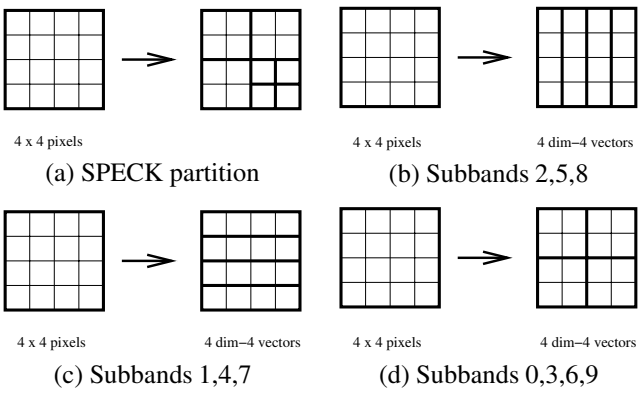


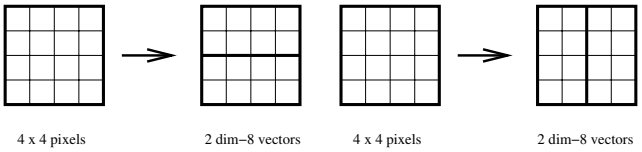
Fig. 1. All subbands.



(a) SPECK partition (b) Subbands 2,5,8

(c) Subbands 1,4,7 (d) Subbands 0,3,6,9

Fig. 2. Vector forming in lower bands



(a) Subbands 10,12,13,15 (b) Subbands 11,14

Fig. 3. Vector forming in higher bands

3. VECTOR SIGNIFICANCE

The basic idea of SPECK is to locate significant coefficients through a set partitioning method and then encode those significant coefficients through scalar quantization. The series 2^n are used as thresholds to determine the significance of a coefficient, starting from $n_{\max} = \lfloor \max_{(i,j)} |c_{i,j}| \rfloor$. If the significant coefficients are to be coded as vectors, a new measure of significance is necessary. Mukherjee and Mitra [3] proposed using the L_2 norm of a vector as the threshold in their vector SPIHT algorithm. A vector $\mathbf{x}$ is significant with respect to n if $||\mathbf{x}|| \geq threshold(n)$. Different sets of thresholds are suggested in their related papers.[3, 4, 5]

In SPIHT or SPECK, a scalar x_i is significant with respect to n when $2^n \leq ||x_i|| \leq 2^{n+1}$. Assuming each element x_i of a vector $\mathbf{x} \in R^4$ satisfies $2^n \leq ||x_i|| \leq 2^{n+1}$, then $2^{n+1} \leq ||\mathbf{x}|| \leq 2^{n+2}$. We choose the thresholds such that a vector $\mathbf{x} \in R^4$ is significant with respect to n, if $2^{n+1} \leq ||\mathbf{x}|| \leq 2^{n+2}$. Although the assumption is not always true, our experiments show that these thresholds perform well enough. By applying the same principle to 8D vectors, a vector $\mathbf{x} \in R^8$ is significant with respect to n, if $2\sqrt{2}2^n \leq ||\mathbf{x}|| \leq 2\sqrt{2}2^{n+1}$.

4. CODEBOOK DESIGN

In SPECK, the value of $n_{\max}$ is sometimes as large as 13 for common test images. The algorithm thus has to start from $n = 13$. The scarcity of such high-energy pixels does not cause problems because no training is required. It becomes an important issue, however, for vectors with large coefficients. Experiments show that coding high-energy vectors generally requires more bits if the distortions are to be within an acceptable range. Experiments suggest that an $n = 10$ codebook is the highest n for which enough data can be collected for training. Such a conclusion also means that any vector significant with respect to $n = 11$ or higher will be encoded by the $n = 10$ codebook.

Reflection on the SPECK algorithm suggests appropriate bit rates for the codebooks. if the target bit rate is 0.2 bpp, then the algorithm stops at $n = 4$ or $n = 5$ so that 7 or 8 bits are allocated to encode significant coefficients located at the first pass of n since the value of $n_{\max}$ is usually 12 or 13. If a 4D vector is formed by these coefficients, using 28 to 32 bits to encode this vector seems a reasonable choice. A codebook with 30 bits is not practical for full-search VQ. A tree-structured VQ such as a multistage VQ, residual VQ, or TSVQ designed by Lloyd splitting is practicable at such codebook sizes, but typically the number of stages is limited to 2 or 3 to avoid the loss of quality due to residual error accumulation. For example, a 2-stage VQ with a 15-bit codebook at each stage can serve as a 30-bit VQ. The optimal bit rate for an $n = 10$ codebook varies among different test images and rate constraints. A multistage tree-structured VQ can be adopted to add more stages so that the bit allocation can be done through successive refinement. For example, the 30-bit codebook can be a 2-stage VQ and each stage codebook is a 3-level tree-structured VQ with 5 bits per level. Such a codebook can be viewed as a 6-stage codebook. Our experiment, however, shows that such multistage codebooks do not improve flexible bit allocation under different rate constraints. Another solution is to design different codebooks with every possible rate. A proper codebook can be selected during a "classified VQ" encoding process based on a test image. We use fixed rate coding for the $n = 10, 9, 8, 7$ codebooks and we prefer full-search VQ to tree-structured VQ designed using a Lloyd splitting algorithm, and we prefer the Lloyd-splitting tree-structured VQ to multistage VQ. We can train as high as a 21-bit tree-structured VQ and a 16-bit full-search VQ in our experimental environment. The candidate codebooks for $n = 10, 9, 8, 7$ are shown in Table 1. Entropy constrained VQ usually performs better than standard VQ with a variable length coder. A codebook with a higher rate can be used as an initial codebook. The average bit rate moves toward the target rate as the Lagrange multiplier λ increases. We can train an ECVQ codebook with a rate of as high as 14 bits in reasonable time. For these reasons, entropy constrained VQ is used only for the $n = 3, 4, 5, 6$ codebooks. The candidate rates are listed in Table 1. As our focus is on low bit rates, we do not consider $n < 3$ codebooks.

We choose $n = 5$ as the highest n value for 8D vectors in the higher subbands based on the distribution of the vectors. All 8D vectors significant to higher n are encoded by the $n = 5$ codebook. The candidate bit rates are determined

by doubling the rate of 4D codebooks with the same n. We designed tree-structured VQ using the Lloyd splitting algorithm with rate 17 to 22 and full-search VQ with rate 15 and 16 for $n = 5$ codebooks. Entropy constrained VQ is used by any codebook with bit rate 14 or below. The list of codebooks is in Table 2.

n	configuration	total rate
10	16_16	32
10	16_15	31
10	15_15	30
10	15_14	29
10	14_14	28
9	13_13	26
9	13_12	25
9	12_12	24
9	12_11	23
9	11_11	22
9	11_10	21
8	12_12	24
8	12_11	23
8	11_11	22
8	11.10	21
8	10.10	20
7	10.9	19
7	9.9	18
7	9.8	17
7	16	
7	15	
7	14	
7	13	
7	12	
7	11	

n	total rate
6	14
6	13
6	12
6	11
6	10
6	9
5	11
5	10
5	9
5	8
5	7
5	6
4	7
4	6
4	5
4	4
3	5
3	4
3	3

'_' indicates a stage in the multistage VQ
'.' indicates a level in the tree-structured VQ

Table 1. 4D codebooks

5. BIT ALLOCATION STRATEGY

Each vector in the significant list is marked by a number n that indicates the norm range of that vector. Mukherjee and Mitra [4] observe that such marking can be viewed as the result of classification. The strategy to select a proper codebook for each class from the candidates is equivalent to an optimal bit allocation problem for the classified VQ. Riskin [6] provided an algorithm to allocate bits optimally for a classified VQ based on the generalized BFOS algorithm. A 2-level tree can be built with the classifier at the root node and each class as a leaf node as in Figure 4.

n	configuration	total rate
5	11.11	22
5	11.10	21
5	10.10	20
5	10.9	19
5	9.9	18
5	9.8	17
5	16	
5	15	
5	14	
5	13	
5	12	

n	total rate
4	12
4	11
4	10
4	9
4	8
3	10
3	9
3	8
3	7
3	6

'.' indicates a level in the tree-structured VQ

Table 2. 8D codebooks

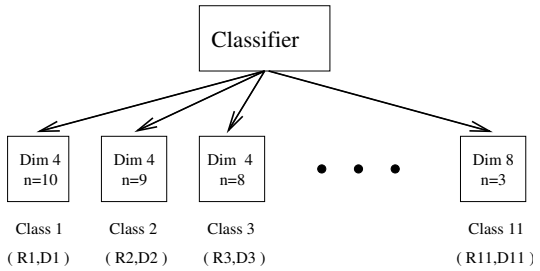


Fig. 4. Codebooks merged into a classified VQ

An input vector is classified at the root node and coded by the codebook of that class. For each class $i = 1, 2, \ldots, M$, the probability is p_i, the average rate is r_i, and the average distortion is $d_i(r_i)$. The overall distortion and rate are

$$\begin{aligned} D &= p_1 d_1(r_1) + p_2 d_2(r_2) + \ldots + p_M d_M(r_M) \\ R &= p_1 r_1 + p_2 r_2 + \ldots + p_M r_M. \end{aligned}$$

The next bit allocation is obtained by pruning class 1 so that $r_1 \Rightarrow r_1'$

$$\begin{aligned} D' &= p_1 d_1(r_1') + p_2 d_2(r_2) + \ldots + p_M d_M(r_M) \\ R' &= p_1 r_1' + p_2 r_2 + \ldots + p_M r_M \\ \lambda &= -\frac{\Delta D}{\Delta R} = -\frac{D' - D}{R' - R} = \frac{d_1(r_1') - d_1(r_1)}{r_1 - r_1'}. \end{aligned}$$

The generalized BFOS algorithm chooses the class causing the minimum distortion increase should be the class to be pruned. We constrain the candidate bit rate for each class to be integers. Therefore $\lambda = d_1(r_1') - d_1(r_1)$ given $r_i - r_i' = 1$.

The complete bit allocation algorithm is

1. Assume each class i has rate $r_i \in (q_{i,1} \ldots q_{i,N_i})$ where $i = 1, 2, \ldots M$.

Set $r_i = q_{i,N_i}$ the maximum rate for each class i.
Add $\mathbf{x} = [r_1, r_2, \ldots, r_M]$ to the output list.

2. Calculate

$$\begin{aligned} S_i(r_i, r_i - 1) &= -\frac{\Delta D}{\Delta R} = -\frac{d_i(r_i) - d_i(r_i - 1)}{r_i - (r_i - 1)} \\ &= d_i(r_i - 1) - d_i(r_i) \\ L &= \arg \min_{i, r_i > q_{i,1}} S_i(r_i, r_i - 1) \end{aligned}$$

Set $r_L = r_L - 1$
Add $\mathbf{x} = [r_1, r_2, \ldots, r_M]$ to the output list.

3. If $r_i = q_{i,1}$ for all $i = 1 \ldots M$, stop.
If not, go to step 2.

The above algorithm generates a set of optimal bit allocations along the rate distortion curve. The best bit allocation can be selected given the operating bit rate. A portion of the bit budget, however, is dedicated to the coding of the significance map in SPECK. It makes the exact number of bits left for the vector quantization hard to estimate. One solution is to select the best bit allocation by trying all candidates. A 6-bit index can be added at the beginning of the output stream to transmit this information.

6. EXPERIMENT

The codebooks are trained by a large corpus composed of 15,000 natural images randomly chosen from the Internet. Each image is cropped into 512×512 pixels and transformed by the 9/7 biorthogonal transform. Such a large corpus should provide enough training data for vectors with high energy. We applied the vector SPECK algorithm to the test images Barbara, Goldhill and Lena and evaluated the PSNR at 0.125, 0.2 and 0.25 bpp. A comparison of SPECK, JPEG 2000 and vector SPECK is given at Table 3. Vector SPECK provides performance that is better than SPECK and comparable to JPEG 2000. The performance improvement is gained at the cost of added encoder complexity due to the vector quantization.

7. CONCLUSION

We have shown that vector SPECK is an effective algorithm for exploiting the correlation between adjacent coefficients. The performance at low bit rate is better than JPEG 2000. The asymmetric property of VQ greatly increases the encoder complexity but keeps the decoder complexity low. Therefore this codec may not be suitable for heavy coding and decoding applications, but it is still a good choice for static applications such image data archiving.

Acknowledgment

The authors thank Professor Pao-Chi Chang for helpful discussions.

Lena			
Bit rate	**SPECK**	**JPEG 2000**	**VSPECK**
0.125	30.96	30.92	31.25
0.2	32.99	32.96	33.47
0.25	34.03	34.09	34.33
Barbara			
Bit rate	**SPECK**	**JPEG 2000**	**VSPECK**
0.125	24.92	25.35	25.36
0.2	26.76	27.17	27.46
0.25	27.68	28.36	28.40
Goldhill			
Bit rate	**SPECK**	**JPEG 2000**	**VSPECK**
0.125	28.38	28.38	28.78
0.2	29.68	29.84	30.11
0.25	30.43	30.53	30.91

Table 3. Comparison of results

8. REFERENCES

[1] A. Said; W.A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, June 1996.

[2] W.A. Pearlman; A. Islam; N. Nagaraj; A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 11, pp. 1219–1235, November 2004.

[3] D. Mukherjee; S.K. Mitra, "Vector set partitioning with classified successive refinement vq for embedded wavelet image and video coding," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1998, vol. 5, pp. 2809–2812.

[4] D. Mukherjee; S.K. Mitra, "Arithmetic coded vector spiht with classified tree-multistage vq for color image coding," in *Proceedings of Second Workshop on Multimedia Signal Processing*. IEEE, 1998, pp. 444–449.

[5] D. Mukherjee; S. Mitra, "Vector spiht for embedded wavelet video and image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 3, pp. 231–246, March 2003.

[6] E.A. Riskin, "Optimal bit allocation via the generalized bfos algorithm," *IEEE Transactions on Information Theory*, vol. 37, no. 2, pp. 400–402, March 1991.