Wavelet Coding Suited For Printer Raster Images

Ricardo L. de Queiroz
XEROX Corporation
800 Phillips Rd. M/S 128-27E, Webster, NY, 14580
queiroz@wrc.xerox.com

ABSTRACT

This paper presents a wavelet-based technique for compression of printer raster images. The coder does not require buffering the whole image and partially takes into account visual losses due to the imaging process by σ-filtering high-pass subbands. All wavelet coefficients are generated, filtered, quantized and entropy-coded sequentially, one block at a time. The coder is suitable for pipeline processing and, for typical images, it shows performance improvements against other schemes in the same class of complexity.

1. INTRODUCTION

In the high-end printing business, documents in a printer pipe are scanned or rendered at high resolutions and compressed for storage. The document images may be pre-collated, reprinted at a later time, processed, and even be exported to another device. Thus, the storage device may be a page buffer, pre-collation memory or a long term storage disk. An outline of the process is shown in Fig. 1. We assume xerographic printers for the moment, so that the documents are to be halftoned before printing. The bitmaps are preserved in continuous tone (more than 1 bit/pixel, bpp) for three basic reasons: (i) printers often have high addressability or analog (line) screens so that the density of dots is much higher than the density of pixels; (ii) the document may undergo image processing operations; and (iii) the document can be exported, edited, or previewed on a monitor.

Desired features for a compression method guided towards a printer raster pipe are:

- good compression ratio for visually lossless compression;
- small image buffer size, avoiding buffering the whole image;
- low complexity for fast software implementation;
- hardware-friendliness;
- avoiding licensing proprietary compression;
- fitting into a “push” or “pull” pipeline architecture.

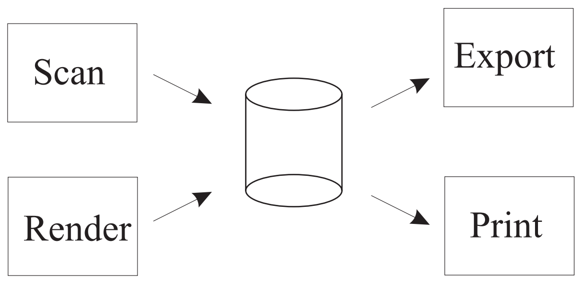

Figure 1. Printing path where image is stored before being halftoned and printed. The export box can comprise an editing, previewing, or transmission system.

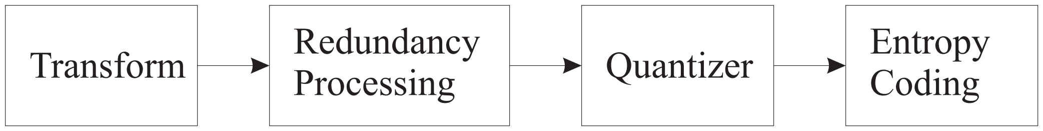

Figure 2. Outline of the coder's basic functions for a non-embedded compressed stream. One block with DWT coefficients is processed and encoded at a time.

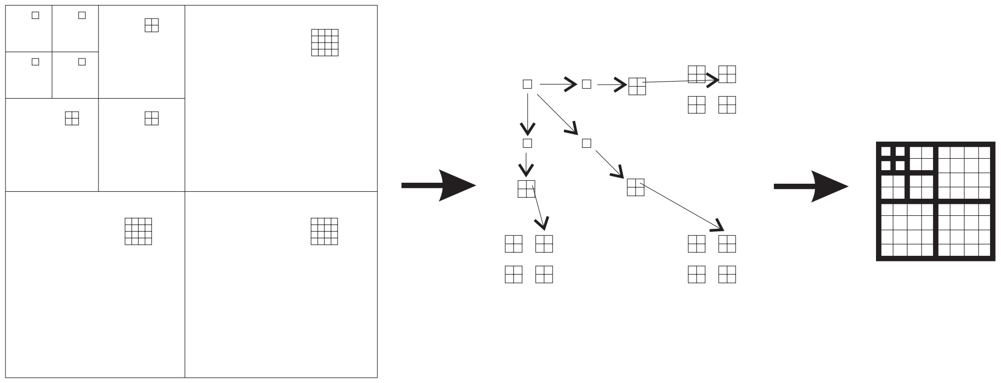

Figure 3. Grouping of DWT coefficients by blocks (similar spatial location) instead of by bands (similar frequency location). The parent-child relationship of DWT coefficients within a block is also illustrated.

The standard JPEG coder[1] is computationally attractive, not requiring much buffering for a pipeline operation. However, JPEG tends to produce more artifacts at lower bit-rates, than more sophisticated wavelet-based coders. Several coders in the literature provide superior compression results compared to JPEG. An example of such a coder is the Said-Pearlman coder or SPIHT,[2] which is related to the EZW coder.[3] Other efficient wavelet and subband coders do exist..[4–12] In general, those coders are not tailored to any specific class of imaging devices, thus, commonly wasting bits to encode information that will not be visible after printing, and they do not comply with most of the desired features listed above.

The intent of this paper is to show a simple wavelet-based coder that meets the above criteria for a printing pipeline. JPEG compression for printing pipelines has been explored[13] assuming the decompressed image is to be immediately halftoned. We do not use such a method because it is not applicable to analog line screens or other types of high-addressability printers.

2. PROPOSED CODER

The compression system outline is shown in Fig. 2, while decompression follows the inverse steps. At each time, only one block of wavelet coefficients is encoded. The basic idea is to serialize operations, while operating at one small tile of the image at a time. The outline in Fig. 2 is similar to the outline of the method used in[12] except by the inclusion of the redundancy processing block, which will be discussed later.

2.1. Discrete wavelet transform

The transform used is a regular discrete wavelet transform (DWT) as is the case in the references. DWT coefficients can be arranged into a tree structure, as is the case in[2,3] and such organization is illustrated in Fig. 3. The DWT coefficients for different bands and same spatial location are arranged into a tree structure. Members of the tree can also be grouped into blocks as depicted in Fig. 3.[12]

Computation and buffering- To compute one block (or one row of blocks) it is not necessary to buffer the whole image. If there are M stages of subband decomposition, there are $3M + 1$ subbands and blocks have size $2^M \times 2^M$. Buffering in the fast scan direction (along image rows) is generally not a problem. However, it is not desirable to store a large number of scan lines (image rows). By designing modules that perform 1-stage 2D decomposition, buffering only the necessary number of rows to produce one low-pass output row and one high-pass output row, the amount of memory required can be greatly decreased.

Independent modules are built and piped, each one able to carry a 1-stage 2D decomposition. The first module in the pipe reads data from the input image while the last one contains a buffer which holds all the coefficients for one row of blocks. The intermediary modules perform the wavelet decompostion. If the filters in the horizontal filter bank have at most L taps each module has to buffer L scanlines of its input. Each intermediary module performs the following tasks:

- receives two new lines of data from its predecessor in the pipe;
- writes the low pass row (one at a time) to the pipe output;
- writes the high-pass samples directly to an output buffer in the the last instance of the pipe.
- discard the two oldest scanlines

Once one row of blocks is computed, it is quantized and encoded.

If the filters have at most L taps, and there are P pixels in an image row, it is necessary to buffer LP samples in the first module $LP/2$ in the second, $LP/4$ in the third, and so on. Thus, all modules together require a buffering capacity of only $LP(2-2^{1-M}) < 2LP$ samples. The output buffer, of course, is required to accomodate $2^M P$ output DWT coefficients.

For a typical 5-stage decomposition using 9/7 filters, the total number of scanlines of DWT coefficients to be buffered is less than 50. At 600 pixels per inch, a page image has dimensions of 6600×5100 pixels. Buffering the DWT of the image would require 33.6M samples. The proposed method, in this example, requires only 255K samples, i.e. an impressive reduction in cost in the order of 132:1.

2.2. Removing invisible information

Assuming the image is to be printed, details are less visible due to two main factors: (i) high resolution imaging yields a high number of pixels per degree of visual angle subtended, and the human visual system is less sensitive to details in these conditions; (ii) the halftone process commonly compromises the reproducible number of gray levels as a function of cell resolution. In this paper, we assume xerographic engines which undergo a halftoning process using high-addressability marking. The idea in our approach is to try to model the printing process losses and to remove the corresponding information. This is not an easy job due to the non-linearity of the marking process and due to the large amount of image processing options available in each printing path. Also, the processing has to be very simple so that it will not significantly affect the coder's complexity.

Commonly, pictures do not need very high resolution imaging and generally 200 pixels/inch (ppi) is satisfactory provided the imaging device can produce enough shades of gray (through high addressability, analog screens, or non-xerographic processes). However, text and graphics do require higher resolution, since the human visual system is very sensitive to edges, mainly text edges. For this reason, any smoothing processing in preparation for the halftoning stage shall keep the edge integrity. For this reason, we settled with a low-complexity approach: the σ-filter.[14] The σ-filter behaves like a regular 2D linear filter, however, it only computes pixels that are within $\pm\sigma$ gray levels of the center pixel. It preserves strong edges while smoothing noisy flat regions. It actually enhances *soft* edges. Therefore, applying the σ-filter to the original image might enhance edges and cause larger DWT coefficients. Our approach is to apply the σ-filter to high-pass DWT bands. In this case, edges are thinned and high frequency noise is removed, while the important detail is preserved, without magnifying coefficients. Other researchers have used a similar concept. For example, morphological filters were used in,[9] edge tracking filters in,[10] and random fields in,[11] among others. In this paper, however, the intent is to eliminate information masked by the printing process.

For tests carried using mid-range Xerox color printers (which have resolution of 400 ppi with an analog line-screen of 200 lines per inch) over several 8bpp test images, it is found that a 3×3 averaging filter with $\sigma = 16$ provides excellent results. In an informal test, when this σ-filter is applied to the three DWT higher frequency bands, it produces printed images which are indiscernible from the original. It is conceivable to use a more aggressive filter, nevertheless we settled for a safer approach as a convenience. Devices with lower addressability such as popular ink-jet printers or higher resolution (such as 600ppi printers with high addressability) may tolerate larger values of σ. In any case, σ can be used as a parameter to tune the algorithm for specific printers.

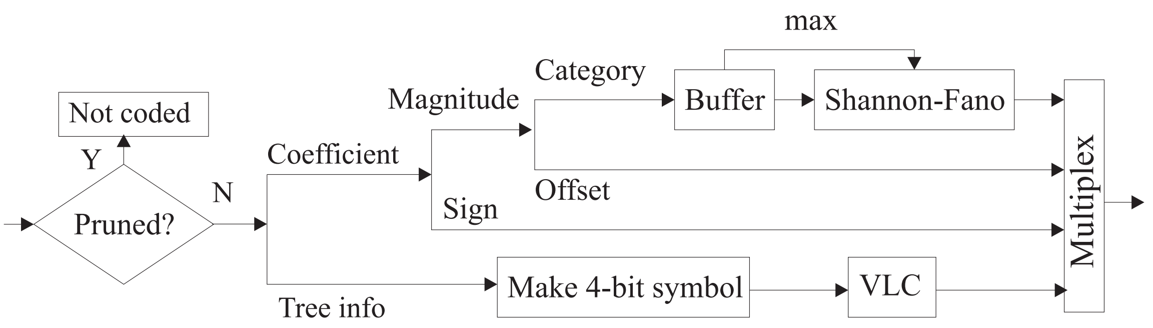

Figure 4. Encoder flow-graph for quantized DWT samples. Information is divided into tree info, sign, category, and offset. Category and tree info are encoded using simple variable length codes (VLC), while sign and offset are not encoded at all. The data is then multiplexed for transmission or storage.

2.3. Quantizing and encoding

All resulting subbands are quantized using uniform quantizers with different step sizes for different bands. The resulting coefficients in a block (tree) of the DWT are scanned into a vector. The vector organization is such that the very low pass band coefficient occupies the first (0) position in the vector. It is also the parent of the three lower frequency bandpass coefficients. If a parent coefficient has index p, the n-th child coefficient ($n = 0, 1, 2, 3$) has index $c = 4p + n$. Similarly, the coefficient for a parent node can be easily found from the address of the child coefficient c, so that $p = c/4$. Thus, cross addressing between child and parent nodes in the tree is facilitated by simple logical bit shift operations:

$$c = p << 2 + n \qquad p = c >> 2$$

The quantized samples in the tree can be lossless encoded using several methods. We denote the resulting coder as raster wavelet coder (RWC).

Note that the coder in[12] also falls into this model, just requiring some vector re-scanning. The set partitioning method for zerotree based coding used in SPIHT[2] can be used to encode quantized coefficients. In this case, the trees (blocks) are fully encoded (all bit planes), one tree at a time. This is possible because the data is pre-quantized. For example, quantizing all bands with quantizer steps which are powers of two and running the set partitioning coder to encode all bit planes is equivalent to stopping the encoding after a fixed number of bit-planes of unquantized integer DWT coefficients. We refer to the coder using this algorithm as SP-RWC. In this paper, both SP-RWC and SPIHT are assumed not to use arithmetic coding.

2.4. Simplified coder

We propose a simplified method for encoding each block which we denote as SC-RWT. In this, the quantized coefficients in the vector are divided into sign, category, offset, and tree information. Encoding is performed in two passes. In the first pass, the leaves of the tree are found, i.e. the shape of the tree is recorded. A node is considered a leaf if all its descendents in the tree have coefficients with zero magnitude. In other words, if all descendents of a node are null they are considered pruned off the tree. In the second pass, the coefficients left in the tree (not pruned) are revisited and encoded following the diagram shown in Fig. 4. Along with the coefficients value, the tree shape information is conveyed by marking and transmitting the position of the leaf nodes.

The tree shape information is encoded in the following way. First, pruned nodes are ignored, leaves are marked as "0" and non-leaves are marked as "1". For the higher frequency bands, node information is not encoded because if a node has not been pruned it is obviously a leaf. Group symbols are formed by grouping every 4 nodes of the tree vector sequence into a 4-bit symbol. The first symbol (whose bits are denoted $B_R B_H B_V B_D$) contains the root node and the nodes for the 3 lowest frequency bands. The root node is never marked as a leaf so that $B_R = 1$. The encoding of this group of nodes is as follows:

$B_H B_V B_D$	code
000	0
111	10
else	$11 B_H B_V B_D$

All other group symbols (whose bits are denoted $B_0B_1B_2B_3$) are encoded in the same manner. Encoding of $B_0B_1B_2B_3$ takes into consideration its *parent* group symbol (denoted as P). The parent symbol acts as a modelling context and the relationship between addresses of parent and child group symbols in a vector is the same as that among addresses of parent and child nodes in a vector. The context conditioning provided by the parent symbol was found to work best when the parent symbol P was all 1's, i.e. if the parent symbol denotes an active region, it indicated that the children are likely either all 0's or all 1's.

$B_0B_1B_2B_3$	code($P = 1111$)
1111	10
0000	11
else	$0B_0B_1B_2B_3$

If P was mixed, the children were found likely to be all 0's.

$B_0B_1B_2B_3$	code ($P \neq 1111$)
0000	1
else	$0B_0B_1B_2B_3$

The category C is the smallest number of bits necessary to represent the magnitude of the coefficient (just like in JPEG[1]), which is the position of the most significant "1" bit in its binary representation. The offset O is the string of $C - 1$ bits necessary to represent the coefficient magnitude, given that we know C and that the leftmost bit is always "1". For example, magnitudes 0 through 5 can be represented as {0,1,10,11,100,101} so that their categories are {0,1,2,2,3,3}. The respective offsets are {∅,∅,0,1,00,01}.

Sign and offset are directly encoded: one bit for sign and $C-1$ bits for offset. The category numbers are encoded using a Shannon-Fano code subject to a maximum number MAX, i.e. C "1"s followed by a "0", except when $C = MAX$ in which case the trailing zero is not necessary. In our implementation, MAX is computed in a block of N consecutive amplitude categories. Category numbers C are grouped into blocks of N samples. MAX is encoded using Shannon-Fano code without maximum, and the other N samples are encoded next. For example, if $N = 8$ and values 0 1 0 1 1 1 1 0 are to be encoded, instead of sending each one using Shannon-Fano code (0 or 10) we encode MAX as 10 and, then, encode each sample as 0 or 1, i.e.:

$$\text{Transmit: } \underbrace{10}_{MAX}\ \underbrace{0\,1\,0\,1\,1\,1\,1\,0}_{\text{no trailing 0}} \quad \text{instead of} \quad 0\ 10\ 0\ 10\ 10\ 10\ 10\ 0.$$

In this simple example, 3 out of 13 bits were saved. It was empirically found that it is always benefitial to encode C in this way and $N = 32$ works best for most images and compression ratios.

3. PERFORMANCE DISCUSSION AND CONCLUSIONS

The coder was tested in several configurations for the compression of 8bpp gray-scale images. In the following presentation, we used a 5-level decomposition, 7/9 bi-orthogonal linear-phase filters and symmetric extensions.[15] Furthermore, we used $\sigma = 16$ and $N = 32$.

In the application of printing raster images, SC-RWC shows advantages over the SPIHT coder because it meets the listed requirements. It requires little buffer space, enough to compute one block of coefficients (because it is not an embeded coder) and it fits into a push or pull pipeline architecture. Furthermore, the coding process is very simple, which requires to visit each coefficient twice at the most. Apart from implementation issues, there are two other points that favor the proposed coder: noise processing through σ-filtering subbands and independent quantization of subbands. However, SPIHT can be modified to incorporate both features. Nevertheless, we compare the proposed coder with SPIHT "as is" (without arithmetic coding) for reference purpose.

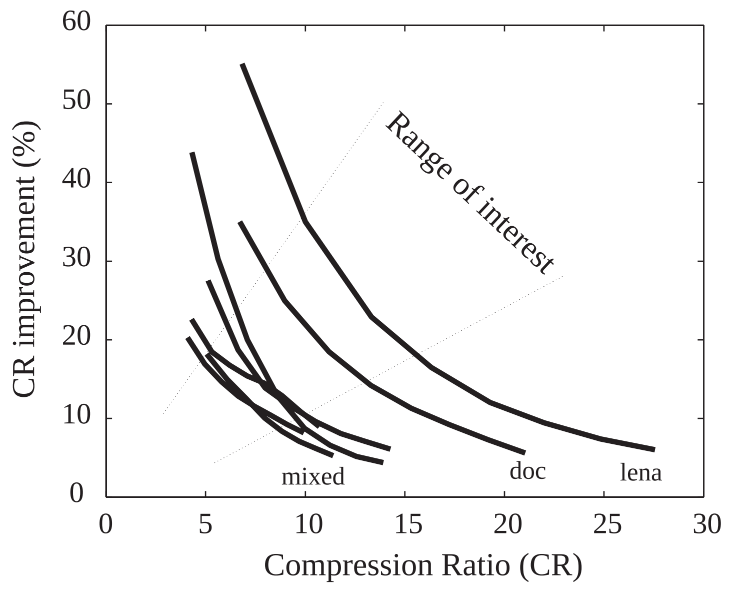

Figure 5. Compression ratio gain as function of compression ratio for several images due to σ-filtering of high-pass bands.

In order to measure the improvement provided by σ-filtering the high-pass bands, we measured the gain in compression ratio (CR) for several images using SC-RWC [*] . Results are shown in Fig. 5. Note that at higher CR, large quantizer steps pick up the task of removing the invisible details and the gain decreases.

To compare SC-RWC and SP-RWC, CR gain computation was also carried for SP-RWC with virtually identical results. Also comparing both methods for the same quantization table, SC-RWC provides less compression than SP-RWC, although the difference is only 3% or less. We consider the difference small enough to be offset by the lower complexity of SC-RWC, which is our preferred coder.

Comparing SC-RWC to SPIHT, we have to keep in mind not only that the σ-filter plays an important role, but also the ability to select independent quantizer steps for different bands. Fig. 5 provides evidence of compression benefits for lower CR values. For high CR values, where the savings from using σ-filter decrease, it is still possible to obtain better looking images by having smaller step sizes for lower frequency bands. This is another form of masking errors. In Fig. 6 we show the original image at the top and two reconstructed images after being compressed at CR=26 using SC-RWC (middle) and SPIHT (bottom). For SPIHT, 6 bitplanes were discarded, which is roughly equivalent to quantizing the bands with the step size of 64. The SC-RWC uses different quantizers, so that the low-pass band has quantizer step of 18, the higher frequency bands have steps of 80, and intermediary bands have intermediary steps sizes. The test image possesses a slow varying *sweep* in light shades. Thus, coarse quantization of low frequency bands can cause *false contours* (*banding*). Also, ringing in the image produced by SC-RWC seems to be less evident than in its SPIHT counterpart.

In conclusion, the SC-RWC is suitable for a printer raster pipeline and only requires very simple predetermined variable-length codes. Our tests have shown that despite its simpler implementation (as compared to SPIHT, for example) the compression loss is minimal and this loss is largely offset by the gains provided by the ability to mask noise in order to save bits (noise processing) and to improve visual quality (different quantizer steps). Tests demonstrate the higher quality of the proposed algorithm for printed images.

[*]It is assumed there is no visual difference between the printed versions of the reconstructed images compressed with and without the σ-filter.

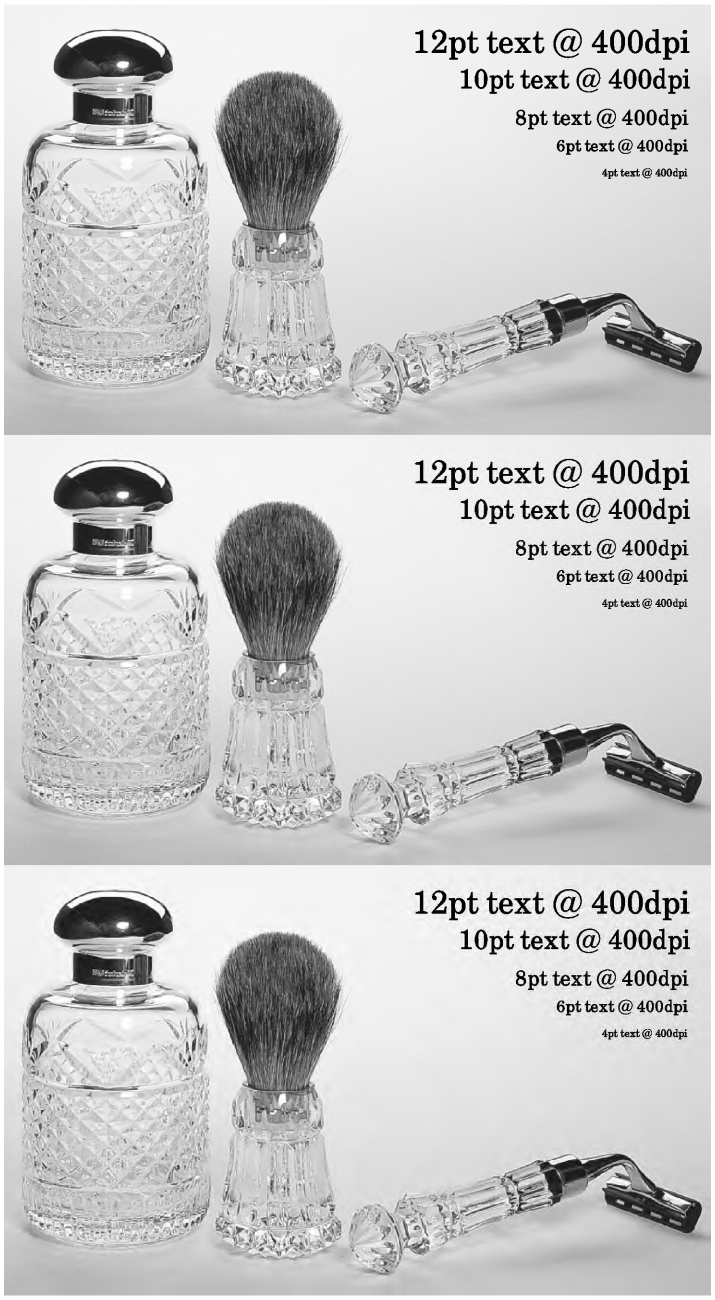

Figure 6. Top: original image. Middle: image compressed using SC-RWC coder at a CR of 26:1, with unequal quantization steps. Bottom: image compressed using the SPIHT coder at 26:1.

REFERENCES

1. W. B. Pennebaker and J. L. Mitchell, "JPEG: Still Image Compression Standard," New York, NY: Van Nostrand Reinhold, 1993.
2. 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, pp. 243–250, June 1996.
3. J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing,* vol. 41, pp. 3445–3462, Dec. 1993.
4. Z. Xiong, K. Ramchandran, and M. T. Orchard, "Space frequency quantization for wavelet image coding," *IEEE Trans. Image Processing,* vol. 6, pp. 677-693, May 1997.
5. J. Li, C.-C. J. Kuo, "Hybrid fractal-wavelet image compression based on a rate distortion criterion," *Visual Comm. and Image Proc. 97,* SPIE Vol. 3024, pp. 1014–1023, 1997.
6. S. LoPresto, and K. Ramchandran, "Wavelet image coding using rate-distortion optimized backward adaptive classification," *Visual Comm. and Image Proc. 97,* SPIE Vol. 3024, pp. 1026-1037, 1997.
7. R. L. Joshi, V. J. Crump, and T. R. Fisher, "Image subband coding using arithmetic coded trellis coded quantization," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 5, pp. 515–523, Dec. 1995.
8. E. A. B. Silva, D. G. Sampson, M. Ghanbari, "A successive approximation vector quantizer for wavelet transform image coding," *IEEE Trans. Image Processing,* vol. 5, pp. 299-310, Feb. 1996.
9. S. Servetto, K. Ramchandran, and M. T. Orchard, "Wavelet based image coding via morphological prediction of significance," *Proc. of IEEE Intl. Conf. Image Processing,* Washington, DC, pp. 530–533, 1995.
10. P. Raffy, M. Antonini, and M. Barlaud, "Zerotree edge-adaptive coder for low bit-rate transmission," *Visual Comm. and Image Proc. 97,* SPIE Vol. 3024, pp. 1067–1076, 1997.
11. J. Luo and C. W. Chen and K. J. Parker, "A scene adaptive and signal adaptive quantization for subband image image and video compression using wavelets," *IEEE Trans. Circuits and Systems for Video Technology,* Vol. 7, pp. 343–357, April 1997.
12. R. de Queiroz, C. Choi, Y. Huh, and K. R. Rao, "Wavelet transforms in a JPEG-like image coder," *IEEE Trans. on Circuits and Systems for Video Technology,* Vol. 7, pp. 419–424, April, 1997.
13. R. A. Vander Kam, P. W. Wong, and R. M. Gray, "JPEG compression for a grayscale printing pipeline," *Proc. of SPIE/IS&T Symp. on Electronic Imaging*, 1995.
14. J. Lee, "Digital image smoothing and the σ-filter", *Computer Vision, Graphics and Image Processing,* vol. 24, pp. 255-269, 1983.
15. G. Strang and T. Q. Nguyen, *Wavelets and Filter Banks,* Wellesley-Cambridge, 1996.