Memory Efficient Quadtree Wavelet Coding for Compound Images [*]

Pamela Cosman, Tamás Frajka, Dirck Schilling, and Kenneth Zeger
Department of Electrical and Computer Engineering, University of California at San Diego
9500 Gilman Drive, San Diego, CA 92093-0407
email: {pcosman, frajka, dschilli, zeger}@code.ucsd.edu

Abstract

Wavelet-based image coders generally perform well on natural images, which are typically characterized by slowly varying image intensities. Their performance suffers, however, on compound images containing both text and image data. We modify a quadtree wavelet coder to perform well on text image data by treating text blocks differently from non-text blocks. We combine wavelet domain processing of non-text blocks with spatial domain processing of text blocks, and achieve improved performance over purely wavelet domain techniques for compound images.

1. Introduction

Text in an image can be far more visually important to a human viewer than might be deduced from summing the energy of the text pixels themselves. Distortion in the edges of text characters, caused by lossy compression of the image, can be more annoying than the same type of distortion in other areas of the image. Unfortunately, wavelet-based image coders suffer from just this deficiency when used on compound images. They often focus on improving low frequency information while allowing high frequency edges, such as the sharp edges of text characters, to blur.

In this paper, we present two variations on a quadtree wavelet-based coder designed for improved performance on compound images. We segment the image to identify blocks containing text, which are then treated specially. One coder operates entirely in the wavelet domain, applying separate coding parameters to text and non-text blocks. In the second variation, the coder combines wavelet-domain processing of non-text blocks with spatial domain processing of text blocks. Both variations provide improved performance over standard wavelet methods when applied to compound images.

This paper is organized as follows. In Section 2 we describe the text segmentation methods used with both coding approaches. In Section 3 we describe the first coder variation, which operates entirely in the wavelet domain. The second variation, combining wavelet and spatial-domain coding, is discussed in Section 4. We present concluding remarks in Section 5.

2 Text Segmentation

Each of the coder variations we describe in this paper begins its processing by segmenting the input image into text and non-text blocks. Any block-based segmenter may be used for this purpose. Our implementation uses a relatively simple procedure based on decision trees.

Training images are divided into 8×8 blocks. For each block, 11 parameters are computed from the 64 pixels in the block. Among these are the row variance, column variance, 3rd and 4th moments, and DCT coefficient energy. The CART (Classification and Regression Trees) algorithm [1] is then used to construct a binary decision tree based on the parameters computed from the training images. Each leaf node of the tree represents either a text or non-text outcome.

At each stage in growing the tree, CART considers which node to split next by considering every parameter at each of the current leaf nodes. The node, parameter, and parameter decision threshold which yield the most accurate partition of the training data are determined, and that leaf node is split. CART grows a large tree, then employs optimal pruning to reduce it to the desired size.

During segmentation, the necessary parameters are computed for an image block. The values are compared with the thresholds in the tree, starting at the top and progressing downward until a leaf node is reached. The resulting text/non-text decision is recorded, and the procedure is repeated on the next block.

Given the 8×8 block size, one bit per 64 pixels would be required to describe the segmentation map. This information is arithmetically encoded using a causal context of four neighbor blocks; it typically adds less than 0.01 bpp to the overall compressed bit rate.

[*] This work was supported in part by the Hewlett Packard Co.

3. Wavelet-Domain Coding of Text Blocks

The quadtree wavelet coder presented in [4] performs well on natural images. Because it uses only one level of wavelet decomposition, and then operates on small blocks, it requires less memory at the decoder compared to zerotree algorithms such as SPIHT [2] and many other wavelet coders.

We modify the coder in [4] to allow separate parameters to be used on text and non-text blocks. The encoder first classifies each 8×8 image block as either text or non-text, and sends this segmentation map to the decoder. Processing proceeds as in [4]. A one-level wavelet decomposition is performed. 8×8 blocks in the low-low (LL) band are processed in scan order. For each block, a "foot" pixel at the lower right corner is predicted from the nearest pixels in the neighboring West and North blocks. The foot prediction error is quantized and sent to the decoder. The block is predicted using bilinear interpolation from the West, Northwest, North, and foot values. If the overall error within the block exceeds a threshold, the block is subdivided into quadrants, and the above procedure is repeated recursively on each quadrant.

In our approach, a LL-band block is considered "text" if all four corresponding spatial blocks were classified as text. Four parameters are selected for each LL-band block depending on its classification as text or non-text:

- The wavelet transform filters. For text blocks, short filters such as Haar are chosen to improve response to sharp edges. The filters are switched on the boundaries between text and non-text regions. To avoid redundancy, at the switching points the pixels of the respective regions are reflected to the other side to provide the right type of pixels for the respective filter taps.
- The quantizer for the foot prediction error. Quantization is finer for text blocks.
- The block error threshold. A lower threshold is chosen for text blocks.
- The quadtree block shape. Text blocks are subdivided into horizontally oriented rectangular subblocks (8×4, 4×2 or 2×1) which were found empirically to perform better than square subblocks in text areas.

The overall effect of this selection is that more bits are invested in text blocks. For any given rate the quality in the text region is improved at the expense of the quality in the non-text regions.

Results: Our simulation results show that treating coefficients corresponding to text blocks differently from non-text blocks can improve (perceptual) image quality. Figure 1 shows the comparison between our proposed method, regular quadtree coding and the Embedded Block Coding with Optimized Truncation (EBCOT) [3] for a portion of the 'cmpnd1' greyscale image at 0.24 bpp. The wavelet domain improved version displayed here uses Haar filters in the text regions with horizontal quadtree block shape and finer quantization. The PSNR values shown are computed over the entire image. As these images demonstrate, PSNR is not a good measure of image quality when it comes to compound images.

4. Combined Wavelet-Domain and Spatial-Domain Coding

Because of the sharp edges and small feature size, text image data is arguably better suited to processing in the spatial domain than in the wavelet domain. In this section we propose an approach which combines wavelet-based coding of non-text regions with spatial domain coding of regions containing text. The algorithm operates at the pixel level; that is, each individual text pixel is coded in the image domain, and all remaining pixels are coded in the wavelet transform domain.

Algorithm Summary: The image is first segmented into text and non-text blocks. The pixels belonging to text blocks are then labeled as either foreground or background (non-text), whereby the encoder chooses for each block which color – light or dark – will be called foreground, and which will be called background. The intensities for the foreground and background colors are then determined for each block. In a segmentation improvement step, some blocks which were previously identified as text blocks can be reclassified as non-text blocks, based on their foreground and background colors as well as those of their neighbor blocks. Following this step, the block segmentation map, the binary foreground/background map for the text blocks, and the quantized foreground intensity values are arithmetically encoded and transmitted.

At this point, both encoder and decoder possess knowledge of the text pixel locations and colors. It is therefore possible to "remove" the text (foreground) pixels to a large degree from the original image, by interpolating their intensities from neighboring non-text pixels. Such a step has the advantage of smoothing the original image, so that it can be more efficiently coded by the quadtree wavelet algorithm. After the text pixels have been removed by interpolation, the remaining image is wavelet-transformed and quadtree coded as discussed in Section 3. In the following sections we describe the steps of this algorithm in greater detail.

4.1 Text Block Color Classification

Within each text block X, the two main colors are identified with Lloyd-Max optimization on a binary scalar quan-

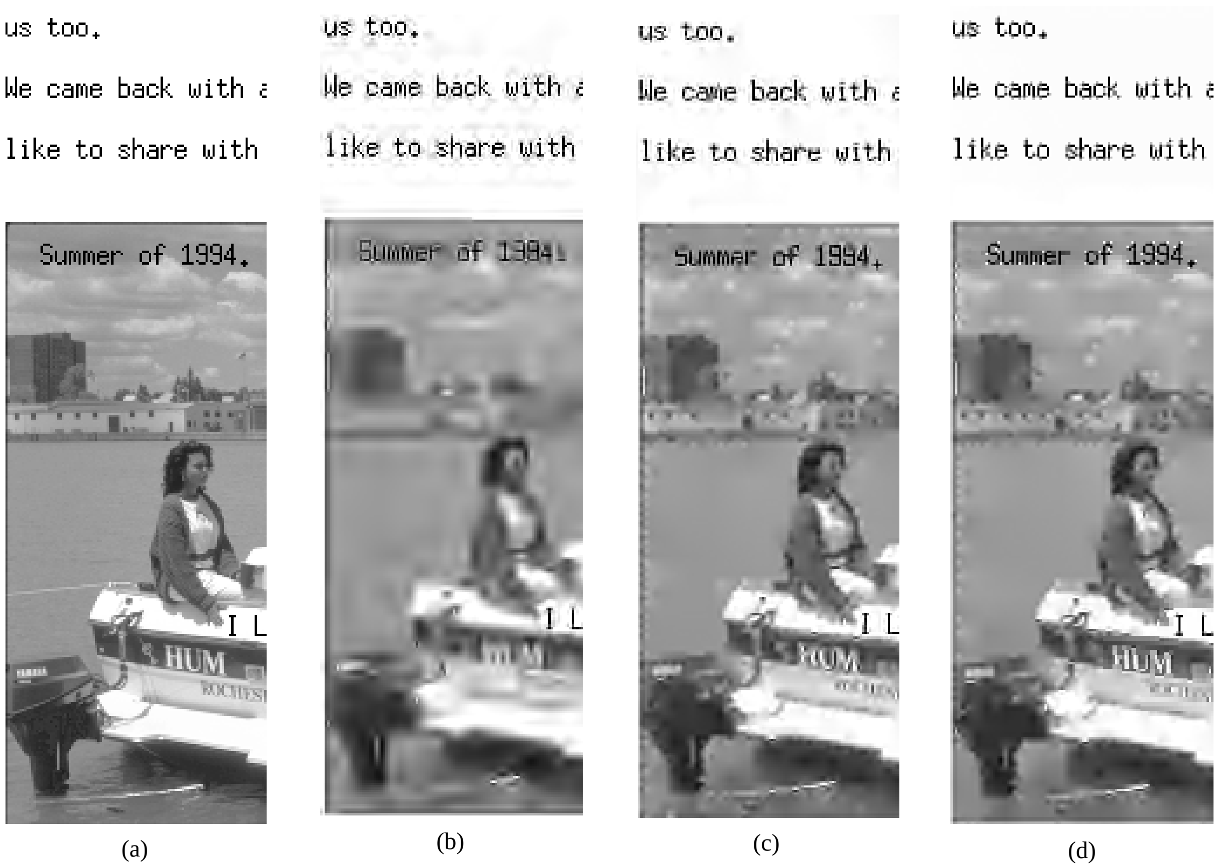


Figure 1. Portions of the ’cmpnd1’ images, (a) original, (b) EBCOT coded at 0.24 bpp, 26.48 dB, (c) Quadtree coded at 0.24 bpp, 25.08 dB, (d) Quadtree coded with improved text region coding at 0.24 bpp, 25.98 dB.

tizer. If the two main colors are closer to each other than some threshold, the block is classified as unimodal and is assigned to be text or non-text based on the foreground and background colors of the neighboring blocks. Otherwise the colors are designated as either foreground or background, based on information about the North and West neighbor blocks. Three cases are considered:

- If both the North and West neighbors are non-text blocks, their average pixel intensity $\overline{I}$ is determined. The color in X which is farthest from $\overline{I}$ is designated as foreground (text), and the other as background.

- If both the North and West neighbors are text blocks, the average of their foreground colors is determined, $\overline{f} = (f_N + f_W)/2$. The color in X which is closest to $\overline{f}$ is designated as foreground, and the other as background.

- If one of the North and West neighbors is a text block and the other non-text, the color in X which is closest to the foreground color in the neighboring text block is designated as foreground, and the other as background. However, if the two foreground colors are far apart, the color in X which is farthest from the background color of the non-text neighbor is designated as foreground.

After the foreground and background colors have been designated, each pixel in block X is assigned to either foreground or background. To avoid distorting the text, only pixels with intensity within a given threshold from the foreground color are assigned to the foreground; the rest are treated as background.

4.2 Map and Foreground Color Coding

The block-level text segmentation map is transmitted as described in Section 2. The foreground/background map for each text block is then arithmetically coded with a causal context of neighboring pixels. Only pixels within text blocks are coded, although contexts may be drawn from neighboring non-text blocks where needed.

The foreground color for a text block X is coded in a predictive coding fashion from the foreground values of the North and West neighbors, exploiting the fact that foreground values don't change much between neighboring blocks.

4.3 Interpolation and Residual Coding of Text Pixels

We now describe the procedure for removing text pixels from the image, in order to allow more efficient wavelet-based coding of the background pixels. The steps of this procedure are illustrated in Figure 2. The original image is shown in profile **A**, with a background area on the left and text on the right. In profile **B**, the text pixel values have been replaced by the block foreground color. In profile **C** the text pixel values are replaced by values interpolated from neighboring background pixels. The image corresponding to profile **C** could be sent directly to the wavelet quadtree coder, and combined by the decoder with the text pixel values transmitted from profile **B** to obtain a reconstructed image. This reconstructed image would be an approximation of profile **B**. The sharp text edges thus obtained can cause the edges of small text characters to appear jagged, however. Instead, we include several additional steps to allow the residual error to be corrected, reducing any jagged appearance.

Profile **D** shows the residual, or difference between the original image and block foreground color in text areas. This residual is added to the interpolated image **C** to yield profile **E**, which is then passed to the wavelet quadtree coder.

At the decoder, a lossy version of **E** is reconstructed, as shown in profile **F**. The decoder also receives the locations of text pixels and the quantized foreground color for each block, which it uses to produce profile **G**. Given the text pixel locations, the decoder interpolates the values of text pixels from neighboring background pixels to obtain profile **H**, which is an approximation of the encoder's profile **C**. The residual **I** is reconstructed as the difference between **F** and **H**. By adding this residual to **G**, the decoder generates the output image profile **J**.

4.4 Ternary Coding of Text Blocks

The procedure described thus far performs well, but further improvement is possible. In many compound images, especially the ones obtained as a result of scanning, the transition from foreground to background color is not sharp; there are some intermediate levels present as well. The discontinuities in the background image introduced by adding the residual can be reduced by more finely quantizing the foreground/background map. Specifically, instead of computing a binary map, we compute a ternary map – i.e., by adding a third "transition" level between "foreground" and "background". This results in smaller residual peaks at the edges of text, which in turn improves the wavelet coding of the background image.

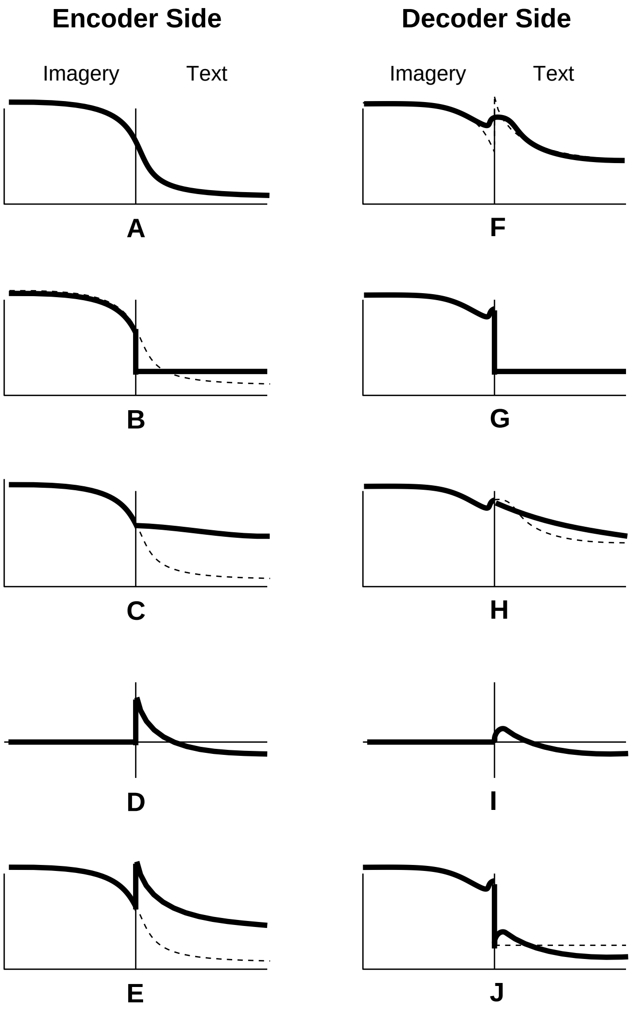


Figure 2. Procedure for removing text pixels from the image to be wavelet-encoded. See description in text.

The ternary procedure is largely identical to the binary procedure described in previous sections. The encoder determines the foreground/transition/background map for each text block in the image. This map is transmitted using ternary arithmetic coding.

The encoder transmits a block's predictive quantized foreground and intermediate color as before. Given these values, the residual encoding procedure is carried out exactly as before. The residual peaks are smaller than in the binary case, however. The image passed to the wavelet coder is thus smoother and can be coded with less distortion at the same rate.

4.5 Results

In Figure 3 we compare results of our combined quadtree wavelet and spatial domain coding approach for color images with the color versions of SPIHT and EBCOT. The original color image was sampled at 100 dpi from a print source. We used the ternary version of our combined wavelet/spatial-domain approach. In the image compressed by our algorithm, text is significantly sharper and clearer than in the SPIHT and EBCOT images. Separate coding of text and image data not only better preserved the text, but the spatial domain coding proved to be more efficient as well, leaving more bandwidth to code the image data. Note that the background edges around the letters "JVC" appear sharper than the same areas in the SPIHT and EBCOT images.

5. Conclusions

These variations on the wavelet quadtree coder improve performance on compound images. While coding entirely within the wavelet domain, one can switch between wavelet filters of different lengths to provide better representation for areas with sharp edges. Text regions can also be handled by mixing wavelet-domain coding of the non-text regions with spatial-domain coding of the text regions. Our spatial domain coding involved an initial binarization or ternarization, and arithmetic coding of the binary or ternary map. However, other types of spatial domain coding could be used instead. The innovations provide superior performance for the sharp edges and small features of text characters, and we expect that they will find other applications as well.

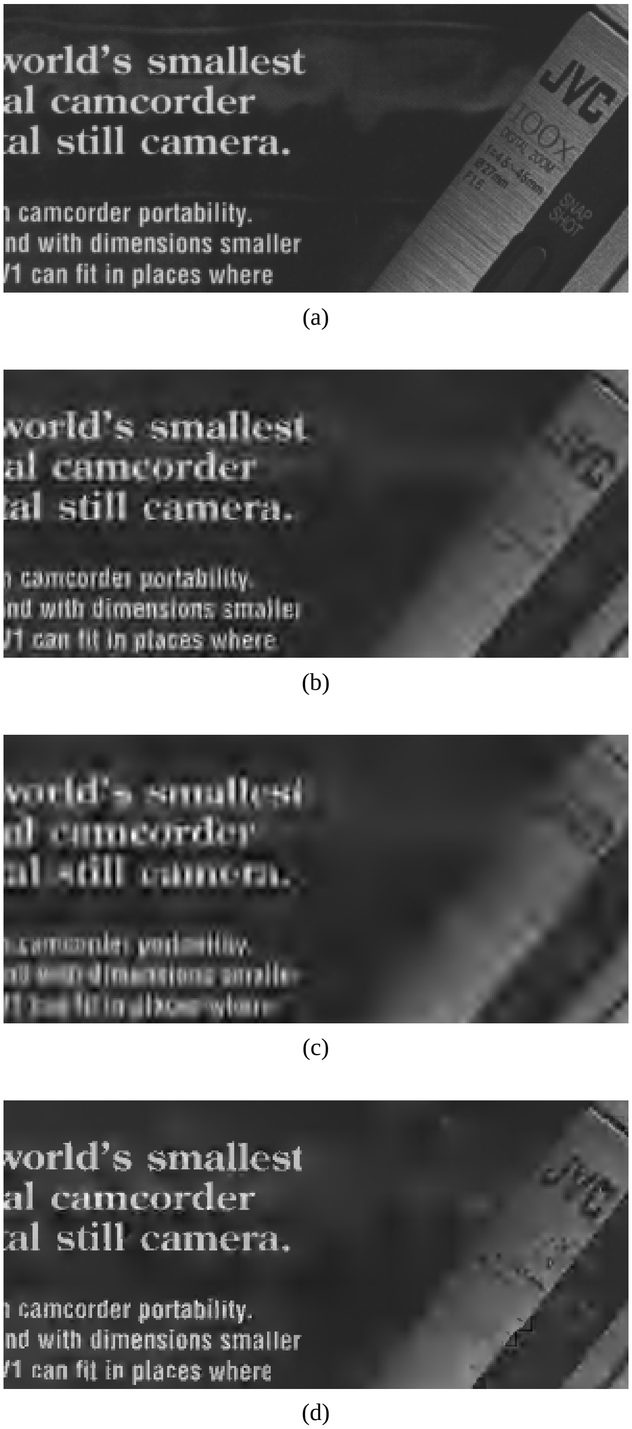
References

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[2] A. Said and W. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(3):243–250, June 1996.

[3] D. Taubman. EBCOT: Embedded block coding with optimized truncation. *ISO/IEC JTC 1/SC 29/WG 1, Document N1020R*, October 1998.

[4] C.-Y. Teng and D. Neuhoff. Quadtree-guided wavelet image coding. In *Proceedings of the 1996 IEEE Data Compression Conference (DCC)*, pages 406–415, Snowbird, Utah, March 1996. IEEE Computer Society Press.



(a)

(b)

(c)

(d)

Figure 3. Portion of the color image 'camcorder', reproduced here in black and white (a) Original image, 100dpi, (b) SPIHT at 0.24 bpp, 27.07 dB, (c) EBCOT at 0.24 bpp, 23.09 dB, (d) Combined wavelet/spatial coding approach, 0.24 bpp, 28.02 dB.