Edge-Enhanced Image Coding for Low Bit Rates

Dirck Schilling and Pamela Cosman
Department of Electrical and Computer Engineering, University of California at San Diego
9500 Gilman Drive, Mail Code 0407, La Jolla, California, 92093
dschilli@ucsd.edu, pcosman@ece.ucsd.edu

Abstract

Many current progressive wavelet-based image coders attempt to achieve the greatest reduction in mean squared error (MSE) with each bit sent. In so doing, they tend to send information on the lowest-frequency wavelet coefficients first. At very low bit rates, images compressed by these coders are therefore dominated by low frequency information and blotchy artifacts. These effects combine to hamper recognition of objects in the images. In this paper, we present a new progressive image coder which employs edge enhancement with the goal of improving the visual appearance and recognizability of compressed images at very low bit rates. Important edges in the original image are captured and transmitted as side information together with a traditional wavelet coder bit stream. The decoder combines the two complementary information sources in a manner which, for certain image classes, can yield highly recognizable images at very low bit rates.

1. Introduction

Recent progressive wavelet-based image coders such as SPIHT [1] and EZW [2] share the characteristic that, at very low bit rates, the decoded image appears blurred and blotchy. By transmitting the wavelet coefficients in strict bit plane order, the coder tends to send low frequency information first. At the very lowest bit rates, the square regions of support of the wavelet bases are clearly visible as stairstep artifacts (blotches) in the developing image. These factors delay recognition by the viewer of the image's content. For many images, however, the information required to identify the image content to a viewer can be represented by a very few lines. An example can be seen in the way a caricature artist is able to create, with only a few strokes, a sketch which is nevertheless instantly recognizable as a specific person. In the same way, a progressive image coder should be able to convey the central meaning of an image within the earliest part of the progression, filling in the remaining details as time and bandwidth permit. The example of the sketch artist suggests that one way to achieve this goal might be by transmitting the primary edge information first.

2. Proposed Approach

Our coder combines SPIHT (or any other progressive wavelet coder) and edge enhancement, with the goal of allowing faster human recognition of the progressively decoded images. A block diagram is shown in Figure 1. The source image is passed through an edge detector, which identifies prominent edges likely to aid in human recognition. Lines are extracted from the identified edge pixels, and the line segment endpoints can be encoded using standard techniques for line graphics, such as modified multiring chain coding. The encoded edge information is sent as part of the image header. Encoding then proceeds in the usual SPIHT fashion. The decoder enhances edges in the output image by combining the edge location information with pixel intensity information available from the progressive SPIHT bit stream.

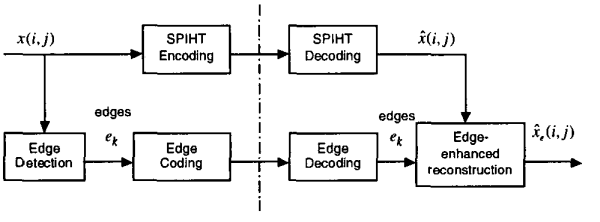


Figure 1: Block diagram of edge enhancing coder.

The coder has the following characteristics:

- The decoder uses the edge information in combination with the SPIHT-encoded bit stream to achieve a natural-appearing result, rather than merely superimposing lines over the developing image.
- Edges are enhanced (sharpened) without obscuring features coming into focus near the enhanced edges.
- Stairstep artifacts in diagonal and curved edges, caused by the rectangular shape of the wavelet bases,

0-7803-5467-2/99/ $ 10.00 © 1999 IEEE

are largely removed in the enhanced edges – unlike with standard edge-sharpening algorithms.

- The compact edge information can be sent in the image header and protected from errors. No further enhancement information must be sent later in the bit stream.

3. Edge Extraction

The edge enhancing progressive coder requires that the edges of interest either be known in advance or detected prior to encoding. The edge enhancement step is independent of the algorithms used for edge detection and line extraction; accordingly, existing, proven methods have been chosen for these portions of the coder. Edge detection is performed by a robust and effective method known as SUSAN (Smallest Univalue Segment Assimilating Nucleus) [3]. This method is based on a nonlinear local region-finding procedure which has several advantages over derivative-based methods such as Sobel or Canny edge detection. These include that it is insensitive to noise, provides good edge localization independently of mask size, and reports sharp corners with good connectivity. Line segment extraction is carried out using methods described by Rosin and West in [4]. The current implementation approximates curved edges by sets of linked straight line segments, however the enhancement technique is applicable to higher-order representations such as arcs, ellipses and polynomials.

The primary goal of our approach is to improve recognition by enhancing only the most "important" edges in the image. There is no widely accepted measure for the importance of an edge, however in most situations a longer edge will convey more information than a shorter one. The edge detection methods described above intrinsically filter out much noise, but still report short, sharply defined edges where they exist. In images containing detailed or cluttered areas, such short edges often occur in closely grouped clusters. Coding these edges is costly, and enhancing them is not likely to improve recognition significantly. We therefore include a further filtering step in the edge detection procedure, which removes edges that are short, have few endpoints, and are distant from other edges.

4. Edge Coding

Numerous methods for encoding line graphics have been described; some of these focus primarily on long continuous curves, long straight lines, or closed contours, cases that are relatively infrequent in natural images. One challenge of the application described here is that the edge information packet must be kept small in relation to the progressive portion of the transmitted stream, which at the bit rates of interest is already tiny. Only a few edges can be transmitted, and most of these consist of very few line segments. There is little information available for adaptive methods to work with.

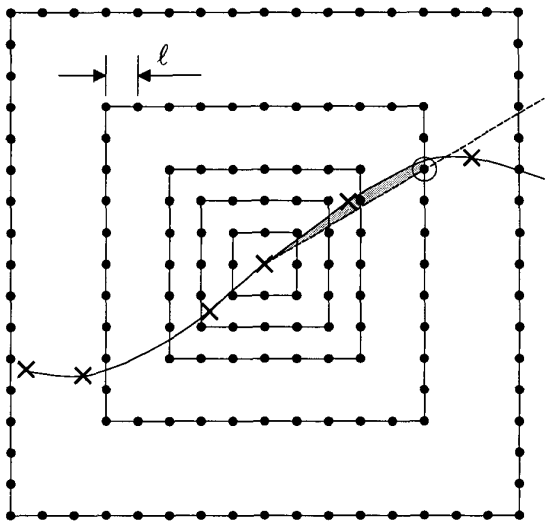


Figure 2: Multiring chain coding grid structure, with the radius 13 ring omitted for clarity. Dots indicate indexed grid points; X's are points on the input curve. The circled grid point indicates the next output point, and the shaded area is the error between the input and output curves for the current step.

4.1 Modified Multiring Chain Code

One efficient method of encoding edge information is the modified multiring chain code with Fibonacci spacing described in [5]. This method was originally proposed for coding human signatures, but can be applied to any form of line-based graphics. The method is summarized in Figure 2. Given a base unit of length ℓ, a structure of concentric square rings is defined whose radii are the product of ℓ and the Fibonacci sequence 1, 2, 3, 5, 8 and 13. Grid points are located along each ring with a spacing of ℓ, yielding a total of 256 grid points. The input curve consists of an ordered sequence of points $X = \{x_o, x_1, \ldots, x_N\}$, which is approximated by the output sequence $Y = \{y_0, y_1, \ldots, y_M\}$. At each step in the procedure, the multiring structure is positioned with its origin on the previously coded output point y_{j-1}. The intersection of the input curve with each ring is examined in succession from the outer ring inward. The nearest grid point to this intersection is chosen as the next candidate output point y_{cand}. If no point along X between y_{j-1} and

y_{cand} is further than ℓ from the line segment (y_{j-1}, y_{cand}), then y_j becomes y_{cand}, and the encoder outputs its multiring index.

This approach encodes the input curve nearly losslessly, in that no point on the input curve is further than ℓ from the approximating output curve. Points on the output curve are a maximum distance of $13\sqrt{2}\ell$ apart, so the choice of ℓ represents a tradeoff between accuracy of the output curve and the number of output points (number of ring indices) required to transmit it. For our implementation, we used $\ell = 1.5$ pixels.

The ring indices are entropy coded to take advantage of the predictability in continuous curves. In our implementation, after each ring index is sent, the grid points are renumbered such that the indices corresponding to "straight ahead" – 0, 8, 24, 48, 88 and 152 – are positioned along the line extending in the direction of the previously encoded line segment. The remaining indices are shifted by a corresponding amount around their respective rings. The result is that each index reflects an angular offset relative to the previously coded line segment, rather than an absolute offset. The distribution of index occurrences is thereby skewed toward smaller angular changes, improving compression of the indices.

4.2 Coding of Start Points and Single Segments

The multiring chain coder must be initialized with the starting point of each edge. For this purpose we employ arithmetic differential offset coding of the edge start points. The edges are first sorted by length. The encoder computes the means of the horizontal and vertical offsets between edge start points, and transmits these along with the total number of edges in an edge packet header. The encoder then arithmetically encodes the differences between the means and the horizontal and vertical offsets to each successive edge from the previous one. Integer arithmetic coding is carried out using techniques similar to those proposed for the JBIG2 standard in [6].

The multiring chain code is particularly effective for long, smooth curves. It performs less well for edges consisting of only one line segment, since no information is available from which to predict the angle of the line segment. For this reason we use differential offset coding to transmit both the start and end points of these edges.

4.3 Edge Coding Performance

Table 1 shows the cost of coding the edge information for several test images. Costs are shown both in total bits, and in *bits per original point* (bpp). Three methods for coding the ring indices are compared: Huffman coding, arithmetic coding with no initial data, and arithmetic coding with initial index probabilities computed from several training images. In both the Huffman and trained arithmetic cases, the training sets used to generate index probabilities did not include the test images. In the arithmetic cases, the coder state included context from prior indices. Arithmetic coding with initial probabilities performed best in most cases except those with the smallest edge packets. Untrained arithmetic coding performed worse than Huffman coding, due primarily to the small amount of data to be coded: the arithmetic coder has too little time to adapt. A disadvantage of the arithmetic technique is that it requires a significant amount of memory at both the encoder and decoder for maintaining state, because of the use of prior index context.

5. Edge Enhancement

We now describe the approach used by the decoder to enhance the edges transmitted in the edge packet. For the current implementation it is assumed that the edges of interest are predominantly of the step or single-sided ramp type. Extensions are possible, however, which allow other types of edges such as narrow ridges to be handled. The basic principle of the edge enhancement procedure is illustrated in Figure 3.

The original edge, shown in profile in part A of Figure 3, is reconstructed by the wavelet coder at a low bit rate as B. The decoder smoothes B to obtain C; alternatively it retains an earlier version of B in memory. The decoder obtains the location of the edge from the image header.

Image	Orig. Edge Points	Huffman bits (bpp)	Untrained Arithmetic bits (bpp)	Trained Arithmetic bits (bpp)
airplane	243	1593 (6.6)	1683 (6.9)	1534 (6.3)
angels	296	2058 (7.0)	2129 (7.2)	1976 (6.7)
apples	1018	5866 (5.8)	5979 (5.9)	5860 (5.8)
aranda	665	3643 (5.5)	3709 (5.6)	3498 (5.3)
bora	386	2335 (6.0)	2486 (6.4)	2388 (6.2)
mtilt	65	963 (14.8)	962 (14.8)	1011 (15.6)
orange	402	2337 (5.8)	2451 (6.1)	2328 (5.8)
picnic	241	1903 (7.9)	1994 (8.3)	1988 (8.3)
wbird	374	2390 (6.4)	2441 (6.5)	2348 (6.3)
wgrass	45	837 (18.6)	790 (17.6)	830 (18.4)
Mean	**373.5**	**2393 (8.43)**	**2462 (8.53)**	**2376 (8.45)**

Table 1: Edge packet coding costs for several images. Figures in parentheses are bits per original edge point.

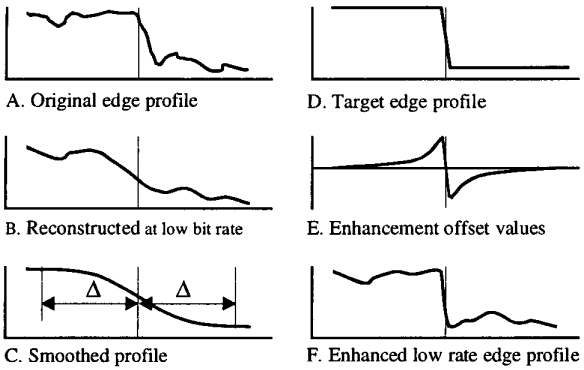


Figure 3: Edge enhancement procedure

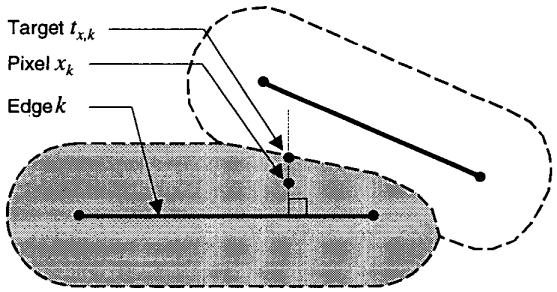


Figure 4: Pixel enhancement for edge k. Pixels in shaded area belong to k.

Based on the intensity values of C at a distance Δ from the edge on either side, a target profile D is computed. In the current implementation, this profile is a nearly ideal step, however other profiles may be chosen. The difference between D and C yields E, the enhancement profile. Finally, E is added to the reconstructed profile B, resulting in the enhanced edge profile F.

In practice, several complications to the above procedure arise, for instance when two or more edges fall within a distance 2Δ of one another. The intent of the target profile D is to determine valid pixel intensities to use for the "high" and "low" sides of the edge. Thus, the desired target pixel intensities should not be drawn from the opposite side of a neighboring edge. To handle this, the decoder computes a distance transform, labeling each image pixel with its distance from the nearest edge and the identity of that edge. An efficient distance transform suitable for this purpose is the 5-7-11 Chamfer algorithm [7]. For each edge, the enhancement procedure is performed on all pixels within Δ of the edge and not closer to another edge. The enhancement target value for a pixel x_k belonging to edge k is drawn from the target pixel $t_{x,k}$, where $t_{x,k}$ is the pixel belonging to k which is farthest from k along the normal line from x_k to k (Figure 4). Examples of the output of our current implementation are shown in Figure 5 and Figure 6.

6. Discussion

In many progressive image transmission applications, such as fast browsing, it is important that the user obtain an understanding of the image contents as early as possible in the progression. The goal of improving recognition at low bit rates may take precedence over that of improving fidelity at higher bit rates. In such applications it is advantageous to place a higher priority on edge information than on low-frequency information when ordering the progressive bit stream. We have described an edge-enhancing image coder which, for certain image classes, allows highly recognizable images to be reproduced at very low bit rates.

Our current work is focused on improving the edge coding in order to limit the edge location information to as small a percentage of the total bit stream as possible. We expect that more sophisticated techniques for removing visually unimportant edges will aid in this goal. We further plan to carry out human observer experiments along the lines of [8] to demonstrate the effect of this approach on image recognizability.

References

[1] 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, 1996.

[2] J. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", *IEEE Trans. on Signal Processing*, 41(12):3445-3462, 1993.

[3] S. M. Smith and J. M. Brady, "SUSAN – A New Approach to Low Level Image Processing", *Intl. Journal of Computer Vision*, 23(1):45-78, 1997.

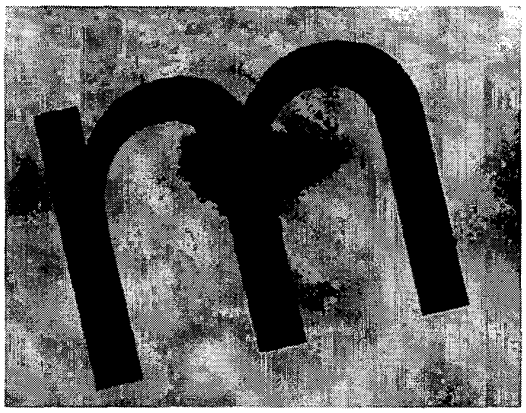
[4] P. Rosin and G. West, "Nonparametric Segmentation of Curves into Various Representations", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 12, 1995.

[5] T. Berger and D. Miller, "Method and an Apparatus for Electronically Compressing a Transaction with a Human Signature", U.S. patent 5091975, 1992.

[6] ISO/IEC JTC1/SC29/WG1 N1093, *JBIG2 Committee Draft*, November 1998.

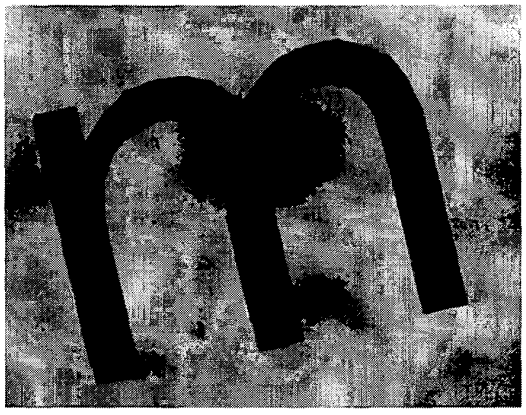
[7] G. Borgefors, "Distance Transformations in Digital Images", *Computer Vision, Graphics and Image Processing*, Vol. 34, No. 3, pages 344-371, 1986.

[8] D. Schilling, P. Cosman and C. Berry, "Image Recognition in Single-Scale and Multiscale Decoders", *Proceedings of the 32nd Asilomar Conference on Signals, Systems and Computers*, Vol. 1, pages 477-481, November 1998.



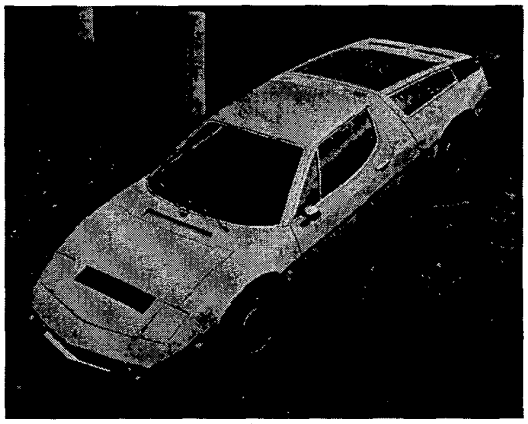
(a)

(b)

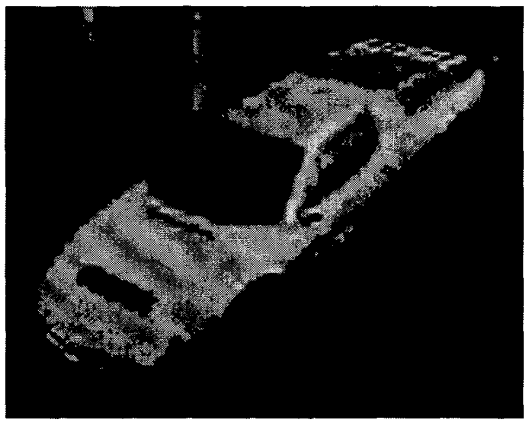


(c)

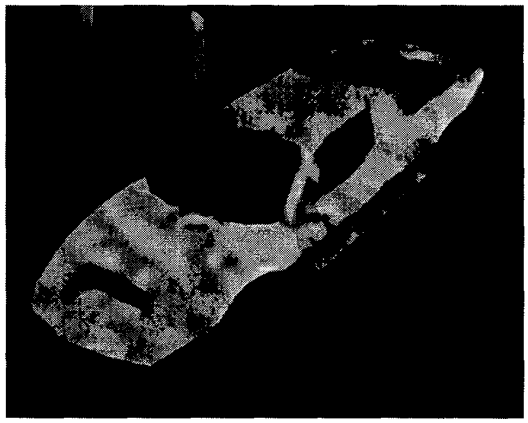
Figure 5: Example of enhancement procedure. (a): Original 512x402 image. (b): SPIHT-coded image at 0.0099 bpp. (c): Edge-enhanced image at 0.0099 bpp. There were 65 edge points. The wavelet coding uses 0.0050 bpp and the arithmetically encoded edge data requires 0.0049 bpp.



(a)



(b)



(c)

Figure 6: Example of enhancement procedure. (a): Original 482x388 image. (b): SPIHT-coded image at 0.0298 bpp. (c): Edge-enhanced image at 0.0298 bpp. There were 386 edge points. The wavelet coding uses 0.0170 bpp and the arithmetically encoded edge data requires 0.0128 bpp.