RATE-DISTORTION OPTIMIZED IMAGE COMPRESSION USING WEDGELETS

Michael Wakin, Justin Romberg, Hyeokho Choi, Richard Baraniuk

Dept. of Electrical and Computer Engineering, Rice University
6100 Main St., Houston, TX 77005

ABSTRACT

Most wavelet-based image coders fail to model the joint coherent behavior of wavelet coefficients near edges. Wedgelets offer a convenient parameterization for the edges in an image, but they have yet to yield a viable compression algorithm. In this paper, we propose an extension of the zerotree-based Space-Frequency Quantization (SFQ) algorithm by adding a wedgelet symbol to its tree-pruning optimization. This incorporates wedgelets into a rate-distortion compression framework and allows simple, coherent descriptions of the wavelet coefficients near edges. The resulting method yields improved visual quality and increased compression efficiency over the standard SFQ technique.

1. INTRODUCTION

Edges are the dominant features in images, with great importance both for perception and for compression. Edges are well known to convey significant information to the viewer, but they have a dramatic impact on compression performance as well: edges account for a significant amount of energy in the frequency domain.

Many of today's leading image coders, ranging from the zerotree (EZW) algorithm [1] to the new JPEG-2000 standard, rely on wavelets to transform and compress the image. Nonetheless, wavelets actually offer inefficient descriptions of edges in images: many wavelet coefficients are required to describe a single edge. A coherency exists among these coefficients which must be preserved during quantization in order to prevent ringing artifacts. Modeling the joint behavior of the coefficients is actually quite difficult, however, and most coders fail to fully capture the joint dependency of wavelet coefficients near edges.

In a sense, edges are simple objects. They can generally be described quite accurately with few parameters, many fewer in fact than the number of pixels (or wavelet coefficients) which are crossed by the edge contour. This inherent simplicity of edges, combined with their importance in terms of compression performance, motivates a search for a geometric tool which can help to efficiently represent the relevant information near edges.

The dictionary of wedgelets developed by Donoho [2] offers one convenient method for describing edges in an image. These dyadic blocks, each containing a single straight edge with arbitrary orientation, can be chained together to approximate an edge contour with desired accuracy. For certain classes of contours, wedgelets have been shown to offer near-optimal nonlinear approximations. Unfortunately, due to errors introduced when approximating real edges with wedgelet step edges, the application of wedgelets in natural image compression is not straightforward. Indeed, wedgelets have yet to yield a complete and viable technique for image compression.

A recent attempt [3] by our group involves coding an image f as a mixture of textures separated by edges. Our approach uses a wedgelet tree (pruned with the CART algorithm [2]) to create a primitive "cartoon" sketch c which represents the dominant edges. After efficiently coding the sketch, we wish to code the residual texture $t = f - c$ using standard wavelet techniques. Unfortunately, residual artifacts created by wedgelet approximations pose many of the same difficulties for wavelets that edges present. A refinement can be made to the residual image [4], masking out any possible artifacts located near edges in the cartoon sketch. This makes the resulting residual image much easier to code using wavelets, and we see a noticeable improvement in visual quality over standard coding techniques. The destruction of information, however, prevents the coder from being competitive in terms of PSNR. Essentially, the use of wedgelets to create a primitive sketch hampers the ultimate efficiency of the coder because the method for residual coding is not simultaneously considered. For a coder with improved PSNR performance, we conclude that the decisions for placing wedgelets must instead be made in a rate-distortion (R/D) framework, so that wedgelets are used only when they actually *improve* the R/D performance of the full coder.

The key to such joint rate-distortion flexibility can be found in the framework of Space-Frequency Quantization (SFQ) [5] coding. The SFQ coder is based on a zerotree [1] quantization framework, with scalar quantization used to compress the significant (non-zerotree) wavelet coefficients. Zerotrees are useful for compressing large collections of low-energy wavelet coefficients using very few bits. The key to the SFQ algorithm is its rate-distortion optimization of the zerotree placements; a tree-pruning operation weighs the rate and distortion consequences of each symbol.

In this paper, we propose an extension of the SFQ method: the addition of *wedgelets* to the R/D optimization. Wedgelets offer some of the same benefits as zerotrees – large collections of wavelet coefficients can be described using very few bits – but wedgelets can do so in the high-energy regions near edges. Section 2 explains the use of wedgelets in describing an edge. Section 3 describes the relevant details of the SFQ coding strategy; the reader is referred to [5] for a complete description of the SFQ algorithm. Section 4 gives the details of our wedgelet-modified SFQ (W-SFQ) algorithm. We note a distinct increase in coding efficiency through the addition of the wedgelet symbols; Section 5 presents an example comparing SFQ and W-SFQ performance. Finally, Section 6 offers conclusions and extensions to our current implementation.

2. WEDGELETS

A *wedgelet* is an $N \times N$ dyadic block that contains a drawing of a single edge with orientation θ and offset d (see Fig. 1(c)). The

This work was supported by the NSF, ONR, AFOSR, DARPA, and the Texas Instruments Leadership University Program.

Email: {wakin, jrom, choi, richb}@rice.edu. Web: dsp.rice.edu.

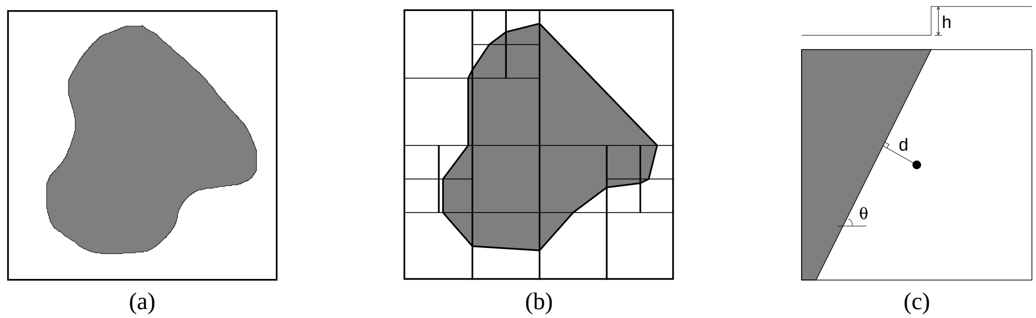

Fig. 1. (a) Artificial image. (b) Wedgelet decomposition. Each dyadic block is constant or contains a single straight edge. (c) Parameterization of a wedgelet on an $N \times N$ dyadic block: θ is the angular orientation of the edge, d is the normal distance from the edge to the center of the block, and h is the difference in average intensities on each side of the edge.

edge separates two regions with grayscale intensities differing by height h. An additional grayscale value (such as one of the two intensities) is also necessary for creating a spatial-domain sketch of the block, but it is not required for our application, which is the prediction of wavelet coefficients.

The *wedgelet dictionary* is the dyadically organized collection of all possible wedgelets. Donoho [2] considers a discrete set of wedgelets for a given block, while we allow continuous parameters. As illustrated in Fig. 1(a)(b), contours in an image may be approximated by a *wedgelet decomposition*: a tiling of wedgelets chosen from this dictionary. In general, long, straight edges are well-approximated using large wedgelets. A series of smaller wedgelets may be required to approximate curved segments of an edge.

For a given $N \times N$ block of pixelized data, several methods exist for obtaining the best wedgelet fit – the set of parameters $(\widehat{\theta}, \widehat{d}, \widehat{h})$ which describe a wedgelet with the closest mean-squared fit to the data. Aside from an exhaustive search over the space $\mathbb{R}^3$ of possible parameters, estimates may also be obtained through an analysis of the Radon transform or the complex wavelet transform [6]. Fast estimation techniques using a discrete wedgelet dictionary, as well as a further discussion of wedgelet analysis, are provided in [7].

3. SPACE-FREQUENCY QUANTIZATION

The SFQ coder [5] is based on a zerotree quantization framework. The dyadic quadtree of wavelet coefficients is transmitted in a single pass from the top down, and each directional subband is treated independently.[1] Each node n_i of the quadtree includes a binary map symbol. A 0 symbol indicates a zerotree: all of the descendants of node n_i are quantized to zero. A 1 symbol indicates that the node's four children are significant: their quantization bins are coded along with an additional map symbol for each. Thus, the quantization scheme for a given wavelet coefficient is actually specified by the map symbol of its parent (or a higher ancestor, in the case of a zerotree); the map symbol transmitted at a given node refers only to the quantization of wavelet coefficients descending from that node. All significant wavelet coefficients are quantized uniformly by a common scalar quantizer; the quantization stepsize q is optimized for the target bitrate.

A tree-pruning operation optimizes the placement of zerotree symbols by weighing the rate and distortion costs of each decision. The pruning starts at the bottom of the tree and proceeds upwards. Initially, it is assumed that all coefficients are significant, and decisions must be made regarding whether to group them into zerotrees. The coder uses several bottom-up iterations until the tree-pruning converges. At the beginning of each iteration, the coder estimates the probability density $p(\widehat{w})$ of the collection of significant coefficients; this yields an estimate of the entropy (and hence coding cost) of each quantized coefficient. Ultimately, adaptive arithmetic coding is used to transmit these quantization bin indices. The SFQ tree-pruning produces a near-optimal configuration of zerotrees without requiring an exhaustive search over all configurations.

Before describing the tree-pruning, we introduce some notation. Let w_i be the wavelet coefficient at node n_i, and let $\widehat{w_i}$ denote the coefficient quantized by stepsize q. The set of the four children of node n_i is denoted C_i, and the subtree of descendants of node n_i is denoted U_i (note that this does not include node n_i).

Optimization in the SFQ framework begins with Phase I, where the tree is iteratively pruned based on the rate and distortion costs of quantization. Phase I ignores the bits required to transmit map symbols, while Phase II adjusts the tree-pruning to account for these costs. In this paper, we focus on Phase I and its adaptation to include wedgelets; the adaptation for Phase II is similar. In each iteration of the Phase I optimization, those nodes currently labeled significant are examined (those already in zerotrees will remain in zerotrees). The coder has two options at each such node: create a zerotree (symbol 0) or maintain the significance (symbol 1). Each option requires a certain number of bits and results in a certain distortion relative to the true wavelet coefficients. The first option, zerotree quantization of the subtree beginning with node n_i, requires $R_i^{(0)} = 0$ bits because no information is transmitted besides the map symbol. This option results in distortion

$$D_i^{(0)} = \sum_{j:\ n_j \in U_i} {w_j}^2.$$

The second option is to send a significance symbol for n_i, as well as the quantization bins corresponding to $\widehat{w_j}$, for j such that $n_j \in C_i$. Note that for this option, we must consider the (previously determined) rate and distortion costs of nodes in C_i as well. Thus

$$R_i^{(1)} = \sum_{j:\ n_j \in C_i} -\log_2 \left[p(\widehat{w_j}) \right] + \sum_{j:\ n_j \in C_i} R_j.$$

[1] Scaling coefficients are coded separately; one approach is mentioned in Section 5.

Fig. 2. Image coded using SFQ optimization. Rate = 0.146 bpp, PSNR = 25.84 dB.

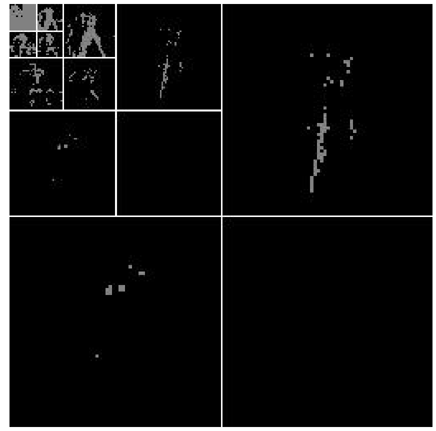

Fig. 3. Multiscale segmentation from SFQ tree-pruning. Zerotrees are represented in black; significant coefficients are gray.

This option results in distortion

$$D_i^{(1)} = \sum_{j:\ n_j \in C_i} (w_j - \widehat{w_j})^2 + \sum_{j:\ n_j \in C_i} D_j.$$

The decision between the two options is made to minimize the Lagrangian cost $J_i = D_i + \lambda R_i$, where λ is the optimization parameter controlling the tradeoff between rate and distortion.

4. WEDGELETS AND SFQ

Despite its success, the SFQ coder fails to model the joint behavior of wavelet coefficients along an edge. A standard SFQ optimization generally results in the use of zerotree symbols to represent smooth regions of the image, with scalar quantization used to code other features such as edges (see Fig. 3). In this section, we propose wedgelets as a third option (symbol 2) in the SFQ tree-pruning. With this extension, wedgelet symbols allow efficient descriptions of the wavelet coefficients surrounding an edge; scalar quantization can be reserved for more complicated texture regions. Moreover, wedgelets implicitly model the joint behavior of the wavelet coefficients, a property that should minimize visual artifacts at low bitrates.

We first describe a method for coding a wedgelet at a node n_i. We then explain how it may be translated into a subtree of wavelet coefficients. Finally, we describe the rate-distortion effects which must be considered during the W-SFQ optimization.

We use a rate-distortion coding framework to transmit a wedgelet block; this operates under the same R/D parameter λ used in Section 3. We code each of the three wedgelet parameters (d, θ, h) using scalar quantization; the quantizer stepsizes are chosen to ensure the correct operating point on the R/D curve. To perform such an optimization, the influence of each parameter's distortion on the squared-error image distortion must be estimated. The height parameter h, for example, is coded first. For large values of h, errors in transmitting d and θ will create significant distortion in the coded wedgelet block; more bits should be used to quantize these parameters. We denote by R_{W_i} the rate required to code all of the wedgelet parameters for node n_i.

Once coded for a node n_i, a wedgelet may be used to predict the wavelet coefficients at all descendants in U_i. This is due to the approximate support of each wavelet basis function within its corresponding dyadic block. One way to obtain a prediction for these coefficients is to create an image containing the coded wedgelet at the appropriate location, take the wavelet transform, and extract the appropriate coefficients. For each j such that $n_j \in U_i$, we denote the predicted wavelet coefficient as $w_{i,j}^*$. Note that this method may actually be used to predict wavelet coefficients in all three subbands; at this time, however, we treat each subband independently.

With an established strategy for coding wedgelet parameters and for predicting the corresponding wavelet coefficients, the addition of wedgelet symbols to the SFQ tree-pruning is straightforward. When a node n_i is transmitted with a wedgelet (symbol 2), the wedgelet parameters are used to compute the values of all coefficients in the subtree U_i. The tree is considered pruned at that point, and no further information is transmitted for any of the node's descendants.

Phase I of the SFQ coder ignored the possible rate cost of transmitting map symbols because ultimately, the cost of 0 and 1 symbols would not differ by a great amount. We expect, however, many fewer wedgelet symbols to be transmitted, and so we find it useful to consider in Phase I of W-SFQ a rough estimate of the probability P_2 (and hence rate cost) of sending symbol 2. Choosing a suitably low value for P_2, it follows that the Phase I rate cost for the wedgelet option is given by

$$R_i^{(2)} = -\log_2 [P_2] + R_{W_i}$$

and the resulting distortion is simply

$$D_i^{(2)} = \sum_{j:\ n_j \in U_i} (w_j - w_{i,j}^*)^2.$$

These costs are weighed against the costs of the other two symbols; the option with the lowest Lagrangian cost is chosen.

As before, the tree-pruning decisions are made at nodes previously designated as significant (symbol 1). Ideally, our W-SFQ implementation would be a true superset of the SFQ coder; that is, every decision made with three symbols should do no worse than the corresponding two-symbol decision from SFQ. We believe, however, that the placement of wedgelet symbols, which prunes a number of coefficients with large magnitude from the significant collection, invalidates the guarantee that the W-SFQ will outperform the SFQ coder. We refer to the next section for evidence of the efficiency of the W-SFQ strategy.

Fig. 4. Image coded using W-SFQ optimization. Rate = 0.146 bpp, PSNR = 25.94 dB.

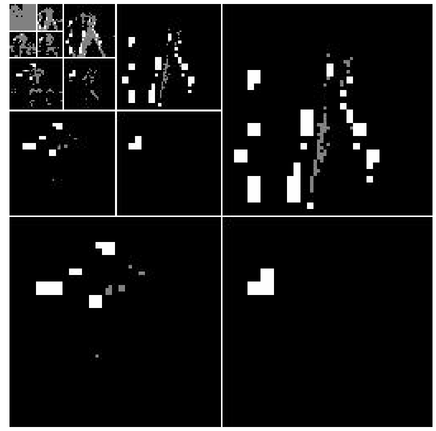

Fig. 5. Multiscale segmentation from W-SFQ tree-pruning. Zerotrees are represented in black; significant coefficients are gray; wedgelet trees are white.

5. RESULTS

For an example of the effectiveness of our W-SFQ extension, we compress the 256×256 *Cameraman* image using both SFQ and W-SFQ. SFQ optimization is designed to compress wavelet coefficients; any efficient technique may be used to separately compress the scaling coefficients. For both SFQ and W-SFQ, we compress the scaling coefficients in a raster scan, predicting each coefficient from its quantized causal neighbors. The prediction errors are quantized and transmitted.

Fig. 2 shows the SFQ compression of the Cameraman image. At a bitrate of 0.146 bpp, a PSNR of 25.84dB is attained. The tree-pruned segmentation is shown in Fig. 3. Black coefficients belong to zerotrees while gray coefficients are significant. The tree-pruning chooses to quantize a total of 3020 significant coefficients. As expected, most of the significant coefficients in the high-frequency subbands occur along edges.

At the same bitrate, Fig. 4 shows the Cameraman image compressed using W-SFQ. A PSNR of 25.94dB is attained, an improvement of 0.10dB over the standard SFQ technique. Fig. 5 shows the tree-pruned segmentation; coefficients in white are described by wedgelets. In this case, a number of previously significant coefficients are now described by wedgelets. The tree-pruning quantizes only 2724 significant coefficients, while transmitting 36 wedgelets at various scales and locations. A total of 1908 wavelet coefficients (small and large) are described in these wedgelet subtrees. Ringing artifacts in the wedgelet-pruned regions are noticeably reduced compared to the SFQ result.

6. CONCLUSION

We have proposed a method for integrating wedgelets into a rate-distortion image compression framework. This technique allows us to take advantage of the simple parameterization of wedgelets, as well as the natural coherency they imply among wavelet coefficients. By extending the SFQ tree-pruning with the addition of a wedgelet symbol, we notice improved visual performance with an increase in PSNR. Presently, though, the usefulness of such an approach is limited to low bitrates and to images containing strong, sharp edges.

Our current research focuses on several improvements which should make wedgelets much more efficient to code, and consequently improve the performance of W-SFQ. Instead of coding wedgelets independently, for example, we could code them in subquadtrees pruned with the CART algorithm [2]. This would help to exploit the dependency among nearby wedgelets. We also believe that a cleverly chosen discrete wedgelet dictionary will help to improve the compression performance by reducing the number of bits required to code a wedgelet, and by helping to model the multiscale dependencies among wedgelet fits [7]. Other possible improvements are mentioned in [8], along with a more thorough discussion of W-SFQ.

7. REFERENCES

[1] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Proc.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[2] D. L. Donoho, "Wedgelets: Nearly-minimax estimation of edges," *Annals of Stat.*, vol. 27, pp. 859–897, 1999.

[3] M. B. Wakin, J. K. Romberg, H. Choi, and R. G. Baraniuk, "Image compression using an efficient edge cartoon + texture model," in *Proc., IEEE Data Compression Conference – DCC '02*, Snowbird, Utah, April 2002, pp. 43–52.

[4] J. Froment, "Image compression through level lines and wavelet packets," in *Wavelets in Signal and Image Analysis*, A. A. Petrosian and F. G. Meyer, Eds. Kluwer Academic, 2001.

[5] Z. Xiong, K. Ramchandran, and M. T. Orchard, "Space-frequency quantization for wavelet image coding," *IEEE Trans. Image Proc.*, vol. 6, no. 5, pp. 677–693, 1997.

[6] J. K. Romberg, H. Choi, and R. G. Baraniuk, "Multiscale edge grammars for complex wavelet transforms," in *IEEE Int. Conf. on Image Proc. – ICIP '01*, Thessaloniki, Greece, Oct. 2001.

[7] J. K. Romberg, M. B. Wakin, and R. G. Baraniuk, "Multiscale wedgelet image analysis: fast decompositions and modeling," in *IEEE Int. Conf. on Image Proc. – ICIP '02*, 2002.

[8] M. B. Wakin, "Image compression using multiscale geometric edge models," M.S. thesis, ECE Dept., Rice University, May 2002.