MULTISCALE WEDGELET IMAGE ANALYSIS: FAST DECOMPOSITIONS AND MODELING

Justin K. Romberg, Michael Wakin, Richard Baraniuk

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

ABSTRACT

The most perceptually important features in images are geometrical, the most prevalent being the smooth contours ("edges") that separate different homogeneous regions and delineate distinct objects. Although wavelet based algorithms have enjoyed success in many areas of image processing, they have significant shortcomings in their treatment of edges. Wavelets do not parsimoniously capture even the simplest geometrical structure in images, and as a result wavelet based processing algorithms often produce images with ringing around the edges.

The multiscale wedgelet framework is a first step towards explicitly capturing geometrical structure in images. The framework has two components: decomposition and representation. The multiscale wavelet decomposition divides the image into dyadic blocks at different scales and projects these image blocks onto wedgelets - simple piecewise constant functions with linear discontinuities. The multiscale wedgelet representation is an approximation of the image built out of wedgelets from the decomposition. In choosing the wedgelets to form the representation, we can weigh several factors: the error between the representation and the original image, the parsimony of the representation, and whether the wedgelets in the representation form "natural" geometrical structure.

In this paper, we show that an efficient multiscale wedgelet decomposition is possible if we carefully choose the set of possible wedgelet orientations. We also present a modeling framework that makes it possible to incorporate simple geometrical constraints into the choice of wedgelet representation, resulting in parsimonious image approximations with smooth contours.

1. INTRODUCTION

Despite their widespread success in image processing, wavelet based algorithms have significant short-comings in their treatment of edge structure in images. Wavelets do not parsimoniously capture even the simplest *geometrical structure* in images. For instance, representing a long, straight edges in images using wavelet basis functions in unduly difficult. Not only does it take many "significant" wavelet coefficients to represent the edge, but those coefficients also have complicated inter-relationships. Disturbing these relationships during processing results in "ringing" around the edges in the final image.

In [1], Donoho introduced the multiscale wedgelet framework as a first step towards explicitly capturing geometrical structure in images. A wedgelet is a piecewise constant function on a dyadic square with a linear discontinuity, see Figure 1. Each wedgelet by itself can succinctly represent a straight edge within a certain region of the image. Smooth contours can be represented by concatenating individual wedgelets from this decomposition. In places where the contour is varying slowly, we can get an accurate representation using a few coarse scale wedgelets (wedgelets on large dyadic blocks). In places where the contour varies more quickly, we use a greater number of wedgelets at finer scales (wedgelets on smaller dyadic blocks). In fact, the wedgelet representation has been shown to have near optimal non-linear approximation [1] and rate-distortion [2] properties for images consisting of piecewise constant regions separated by smooth boundaries.

There are two components in the multiscale wedgelet framework: decomposition and representation. The *multiscale wedgelet decomposition* (MWD), discussed in detail in Section 2, divides the image into dyadic blocks at different scales and projects these image blocks onto wedgelets at various orientations (an example of an image block being projected onto a wedgelet is shown in Figure 2). In Section 3, we present a fast algorithm for calculating the MWD of an image. The algorithm builds up projections onto wedgelets at coarse scales using previously calculated projections at finer scales.

A *multiscale wedgelet representation* (MWR) is an approximation of the image I built out of wedgelets from the MWD. A MWR is constructed by choosing a set of dyadic squares (not necessarily at the same scale) that partition $[0,1]^2$ and a wedgelet contained in each, see the example in Figure 6. Calculating the MWR amounts to solving a regularized optimization problem with several factors being weighed against one another: the L^2 error between the representation and the original image, the parsimony of the representation (complexity constraints), and whether the wedgelets in the representation form "natural" geometrical structure (geometry constraints). In Section 4, we discuss how to use scale-to-scale dependencies between MWRs at different resolutions to ensure smooth contours (like we expect in real-world images) in the multiscale wedgelet representation. The structure of the model allows efficient algorithms for finding the optimal MWR given an image and a set of complexity and geometry constraints.

A major strength of the wedgelet framework (and the closely related beamlet framework in [3]) is that it captures geometrical structure of the image at multiple scales. We can infer the gross geometrical structure of an image using a coarse (parsimonious) wedgelet representation. Refining the wedgelet representation not only yields a more accurate image approximation (in terms of squared-error), but results in a better geometrical description as well.

Applications of wedgelet decompositions and representations include detecting linear singularities in the presence of noise [3]

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

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

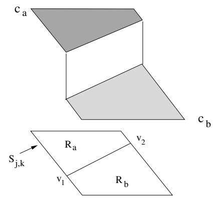

Fig. 1. *A wedgelet on a dyadic square $S_{j,k}$ is a piecewise constant function over two regions R_a and R_b on either side of the line defined by the orientation (v_1, v_2).*

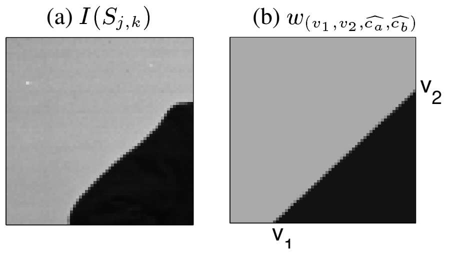

Fig. 2. *(a) A section of the "Cameraman" image on a 64×64 dyadic square $S_{j,k}$. (b) Projection of $I(S_{j,k})$ onto wedgelet with orientation (v_1, v_2).*

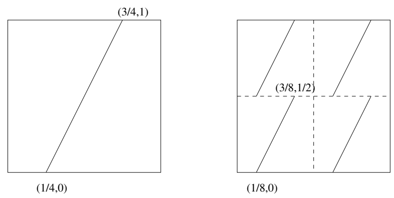

Fig. 3. *A wedgelet at $((\frac{1}{4}, 0),\ (\frac{3}{4}, 1))$ in the coarsest scale dyadic block has the same relative orientation as a wedgelet at $((\frac{1}{8}, 0),\ (\frac{3}{8}, \frac{1}{2}))$ in the "lower left" dyadic block at the next finest scale.*

and image coding [4]. The contributions of this paper are the efficient discrete multiscale wedgelet decomposition discussed in Section 3, and the use of wedgelet representations as a framework for geometry modeling, discussed in Section 4.

2. WEDGELET DECOMPOSITIONS

A *wedgelet* w is a function on a square S that is piecewise constant on either side of a line ℓ through S. Four parameters are needed to define w: two parameters for ℓ — in this paper we'll use the points (v_1, v_2) where ℓ intersects the perimeter of S — and the values w takes above (c_a) and below (c_b) ℓ (see Figure 1). The (v_1, v_2) determine the *orientation* of w, while c_a and c_b determine its *profile*, i.e. the size and offset of the "jump" in the wedgelet. A function that is constant over all of S is called a *degenerate* wedgelet (we can think of such a function as a wedgelet where the line ℓ does not pass through S). We will use $w_{(v_1,v_2,c_a,c_b)}$ to denote a wedgelet when we want to make the parameterization explicit.

Let I be an image[1] supported on $[0, 1]^2$ and $S_{j,k} \subset [0, 1]^2$, $k = \{k_1, k_2\}$ be a dyadic block at scale $j \in \mathbb{Z}, S = [k_1/2^j, (k_1 + 1)/2^j] \times [k_2/2^j, (k_2 + 1)/2^j]$ for integers $0 \le k_1, k_2 < 2^j$. The wedgelet decomposition $\mathcal{W}(I(S_{j,k}))$ of an image I over $S_{j,k}$ is a collection of projections of $I(S_{j,k})$ onto wedgelets at a finite set of orientations $\mathcal{V}$. For each orientation $(v_1, v_2) \in \mathcal{V}$, $S_{j,k}$ is divided into two regions R_a (the region above the line ℓ defined by (v_1, v_2)) and R_b (the region below ℓ). The profile of the wedgelet at orientation (v_1, v_2) is calculated by averaging $I(S_{j,k})$ over these regions:

$$\widehat{c_a} = \text{Average}(I(S_{j,k})|R_a) \quad (1)$$
$$\widehat{c_b} = \text{Average}(I(S_{j,k})|R_b). \quad (2)$$

See Figure 2 for an example. Collecting the wedgelets into a set, we get

$$\mathcal{W}(I(S_{j,k})) = \{w_{(v_1,v_2,\widehat{c_a},\widehat{c_b})} : \ (v_1, v_2) \in \mathcal{V}\}. \quad (3)$$

We can infer the local geometrical structure of I in the region $S_{j,k}$ by finding the orientations for which $w_{(v_1,v_2,\widehat{c_a},\widehat{c_b})}$ is a good approximation to $I(S_{j,k})$. If $I(S_{j,k})$ has a discontinuity along a linear edge, then the error between $I(S_{j,k})$ and $w_{(v_1,v_2,\widehat{c_a},\widehat{c_b})}$ will be small for the orientation (v_1, v_2) closest to that edge.

The multiscale wedgelet decomposition $\mathcal{W}^J(I)$ of an image I is the collection of wedgelet decompositions $\mathcal{W}(I(S_{j,k}))$ for all dyadic squares $S_{j,k}$,

$$\mathcal{W}^J(I) = \{\mathcal{W}(I(S_{j,k})) : \ j = 0, \ldots, J, \ k_1, k_2 = 0, \ldots, 2^j\}. \quad (4)$$

In this paper, we will assume that the set of wedgelet orientations is the same *relative* to each dyadic square. For example, if we project onto a wedgelet at orientation $((\frac{1}{4}, 0),\ (\frac{3}{4}, 1))$ at the coarsest scale dyadic block, we will project onto a wedgelets at $((\frac{1}{8}, 0),\ (\frac{3}{8}, \frac{1}{2}))$, $((\frac{5}{8}, 0),\ (\frac{7}{8}, \frac{1}{2}))$, $((\frac{5}{8}, \frac{1}{2}),\ (\frac{7}{8}, 1))$, and $((\frac{1}{8}, \frac{1}{2}),\ (\frac{3}{8}, 1))$ at the four dyadic blocks at the next finest scale (see Figure 3). In the sequel, we will label wedgelet orientations as they would be labeled on $[0, 1]^2$; the four orientations shown in Figure 3 would all be labeled $((\frac{1}{4}, 0),\ (\frac{3}{4}, 1))$.

For an $N \times N$ pixel image, calculating a complete multiscale wedgelet decomposition $\mathcal{W}^J(I)$ with $J = \log_2 N$ has $O(MN^2 \log_2 N^2)$ computational complexity, where $M = |\mathcal{V}|$ is the number of wedgelets for each dyadic square. In the next section, we will show that by a judicious (but still diverse) choice of orientations (v_1, v_2), we can calculate $\mathcal{W}^J(I)$ with $O(MN^2)$ complexity.

3. A FAST MULTISCALE WEDGELET DECOMPOSITION

Calculating the multiscale wedgelet decomposition requires computation of projections (see equations (1) and (2) above) for M different wedgelets in each dyadic square $S_{j,k}$. In general, the number of operations for each projection is proportional to $N^2 2^{-j}$, the number of pixels in $S_{j,k}$. We can reduce the total number of operations in the wedgelet decomposition $W(I(S_{j,k}))$ by "building up" projections at coarse scaled using projections at finer scales.

Figure 3 gives an example of using wedgelet projections at a finer scale to compute a wedgelet projection at a coarse scale. The wedgelet at orientation $((\frac{1}{4}, 1),\ (1, \frac{1}{4})$ on a dyadic square breaks down into four parts at the next scale:

[1] I is, in general, a function of a continuous variable on the unit square $[0, 1]^2$. For discrete $N \times N$ pixel images, we can think of I being piecewise constant over squares with sidelength $1/N$.

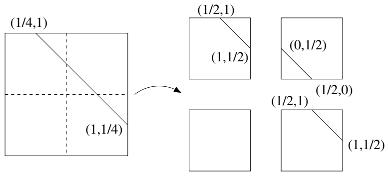

Fig. 4. *A wedgelet at orientation* $((1/4, 1),\ (1, 1/4)$ *can be broken down at the next finest scale into three wedgelets with different orientations and one constant region.*

1. A wedgelet of orientation[2] $((\frac{1}{2}, 1),\ (1, \frac{1}{2})$ in the "upper left" dyadic subsquare.
2. A wedgelet of orientation $((0, \frac{1}{2}),\ (\frac{1}{2}, 0)$ in the upper right dyadic subsquare.
3. A wedgelet of orientation $((\frac{1}{2}, 1),\ (1, \frac{1}{2})$ in the lower right dyadic subsquare.
4. A constant on the lower left dyadic subsquare.

Given these finer scale wedgelet profiles, it is straightforward to calculate the desired wedgelet profile at the coarse scale.

To calculate a wedgelet projection in this manner, we need to choose our set of orientations $\mathcal{V}$ so that wedgelets at every orientation $(v_1, v_2) \in \mathcal{V}$ break down into other wedgelets at orientations in $\mathcal{V}$. Consider the set of orientations with intersection points equally spaced at distance $1/q$, $q \in \mathbb{Z}$ around the perimeter of $[0, 1]^2$. Which wedgelet orientations (pairs of points on the perimeter) will allow us to calculate projections by using results at a finer scale? A quick example will illustrate that not all wedgelet orientations are *admissible*:

> Take $q = 4$ and consider the wedgelet w at orientation $((\frac{1}{4}, 0),\ (1, \frac{1}{2}))$. At the next finest scale, w breaks down into two wedgelets at orientations $((\frac{1}{2}, 0),\ (\frac{1}{3}, 1))$ and $((0, \frac{1}{3}),\ (1, 1))$, neither of which are in $\mathcal{V}$.

The condition for an admissible orientation (v_1, v_2) depends on the slope λ of the line defined by (v_1, v_2). The orientation is admissible if $\lambda = 0$, $\lambda = \infty$ (horizontal and vertical orientations are admissible), or if $\lambda = a/b,\ a, b \in \mathbb{Z}$ is rational and a and b divide q evenly. As an example, the admissible orientations for $q = 4$ — there are $M = 56$ total, with lines of slope $0, 1/4, 1/2, 1, 2, 4$, and ∞ — are shown in Figure 3. Note that $\mathcal{V}$ is fairly diverse; there is an orientation in $\mathcal{V}$ "close" to every possible orientation.

In this paper, we have just considered wedgelets that can be built out of other wedgelets at the *next* finest scale. It is possible to get a richer set of orientations by building up coarse scale wedgelets using wedgelets at *all* finer scales.

If every orientation in $\mathcal{V}$ is admissible, then calculating a wedgelet projection is extremely fast; it simply requires the addition of four numbers. This reduces the computational complexity of a full MWD $\mathcal{W}^J(I)$, $J = \log_2 N$ to $O(MN^2)$.

The multiscale wavelet decomposition $\mathcal{W}^J(I)$ gives us a number of different ways of describing the image I. We can use any one of M (possibly degenerate) wedgelets to describe each dyadic block of the image. In the next section, we discuss how to select a set of wedgelets that best describes an image.

[2] Remember, wedgelet orientations are labeled relative to $[0, 1]^2$; see Section 2.

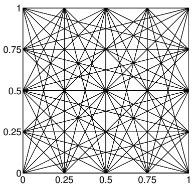

Fig. 5. *Set of admissible wedgelet orientations for* $q = 4$. *By restricting ourselves to these wedgelets in each dyadic square, we can compute a multiscale wedgelet decomposition efficiently.*

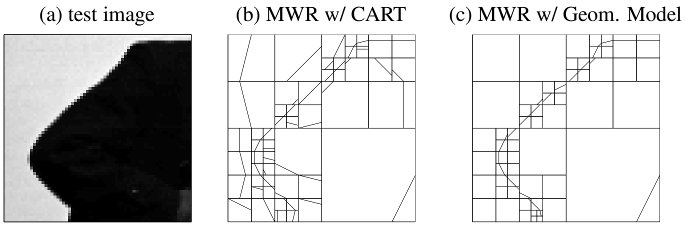

Fig. 6. *Wedgelet orientations used in the multiscale wedgelet representation of a test image (a). (b) Orientations found with CART and (c) with a geometry model via dynamic programming.*

4. MODELING MULTISCALE WEDGELET REPRESENTATIONS

Once we have calculated the projections in the MWD $\mathcal{W}^J(I)$, we can use the results to choose a representation of the image I. A *multiscale wedgelet representation* (MWR) W consists of a set of (possibly different sized) dyadic squares that partition $[0, 1]^2$, each supporting one (possibly degenerate) wedgelet[3] — see Figure 6 for an example. The projection of a region of I onto the wedgelet the wedgelet in a dyadic square gives us geometrical information about that region of th

Given an image, the choice of the MWR W can be optimized over multiple criteria. Of particular interest to us is

Approximation We want the MWR to be a close approximation (in the L^2 sense) the image. As the dyadic squares used in the MWR to partition $[0, 1]^2$ become smaller (increasing the resolution of the MWR), the representation becomes more accurate.

Parsimony We want to describe the image using the smallest number of wedgelets possible. As the dyadic squares in the MWR become smaller, more wedgelets are being used to approximate the image.

Geometry The wedgelets in the MWR should form natural geometrical structures, e.g. smooth, connected contours.

Of course, these are not the only criteria that exist. For instance, in [4], the authors propose an algorithm that chooses a wedgelet approximation that maximizes coding gains.

In [1], the CART algorithm is used to find a MWR with an optimal tradeoff between the Approximation and Parsimony criteria. Given a weighting parameter λ, we can exploit the quadtree structure of the representation to find a fast solution to the optimization

[3] We will find it useful to think of a MWR as a pruned quadtree with a wedgelet living at each leaf.

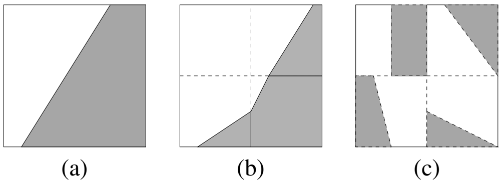

Fig. 7. *When the coarse scale wedgelet representation in (a) is resolved into subblocks, we expect the wedgelet fits at the next scale to refine the geometry of the image as in (b), and not be completely arbitrary as shown in (c).*

problem

$$\min_W \mathrm{D}(W) + \lambda|W| \tag{5}$$

where $\mathrm{D}(W)$ is the mean-square error between the image I and the MWR W, and $|W|$ is the number of terms (leaves of the quadtree) in W.

In this section, we will discuss ways to incorporate geometry into the selection of the MWR by formulating a problem similar to (5). By imposing a geometry model for smooth contours using the relationships between MWRs of increasing resolutions, we will be able to quantify how well a particular arrangement of wedgelets fits our notion of edge structure. We can characterize the way we expect contours to behave in an image by imposing dependencies between the wedgelet *orientations* in the different dyadic blocks of the MWR.

To capture these dependencies, our geometry model will describe how we expect the orientations in the MWR to change as we increase the resolution (making the MWR less parsimonious but a better approximation to I). For example, Figure 7(a) shows the best wedgelet fit on a particular dyadic square. When we refine the estimate, breaking the dyadic square into four subsquares, we would expect the best wedgelet fits to be oriented as shown in Figure 7(b), and would not expect them to be oriented as in Figure 7(c).

The inter-scale relationships between the wedgelet fits can be captured using a quadtree structured finite Markov model. The state at each node represents the orientation of the best wedgelet fit in the corresponding dyadic square. A state transition matrix is used to score different wedgelet refinements. For example, the transition from the parent state in Figure 7(a) to the child states shown in Figure 7(b) would receive a high score, while the transition to the child states shown in Figure 7(c) would receive a much lower score.

The most practical model is to make each of the four "child" orientations independent given the "parent" orientation. Then the model is specified by an $M \times M$ transition matrix whose entries are the conditional probabilities of the child orientation given the parent orientation.

The probabilities are assigned based on the distance[4] $d(\ell_{\mathrm{parent}}, \ell_{\mathrm{child}})$ between the lines ℓ_{parent} and ℓ_{child} that define the parent and child wedgelet orientations. For the examples in this paper, we use $\Pr(\ell_{\mathrm{child}}|\ell_{\mathrm{parent}}) \sim e^{-d(\ell_{\mathrm{parent}},\ell_{\mathrm{child}})^2}$.

Now that we have the geometry model, we can use it to regularize our choice of MWR. We can again exploit the quadtree structure of the wedgelet representation along with the Markov nature of the model to efficiently find an exact solution to

$$\min_W \mathrm{D}(W) + \lambda[-\log_2 \mathrm{P}(W) + |W|] \tag{6}$$

where $\mathrm{D}(W)$ and $|W|$ are the same as in (5) and $\mathrm{P}(W)$ is the likelihood of W under our geometry model. The dynamic program that solves (6) is similar to the Viterbi algorithm in communications.

We can also interpret (6) as a rate-distortion optimization problem: the $-\log_2 \mathrm{P}(W)$ tells us the number of bits it would take to code each wedgelet orientation in the chosen W using our geometry model, with an additional $|W|$ bits to indicate which of the nodes are pruned.

Figure 6(c) shows an example of using the geometry model to choose the wedgelet orientation. Notice that the local contour information is much "cleaner" than that found by the CART algorithm; incorporating a geometry model allows us to be prudent about the kind of structure we extract from an image.

5. CONCLUSIONS

The inadequacy of wavelets in representing contours motivates the need for decompositions that explicitly take advantage of the geometrical regularity in images, i.e. that these contours are in general slowly varying, to provide a sparse representation. The multiscale wedgelet decomposition is one such representation. In this paper, we have shown that the quadtree structure of the MWD allows us to:

(a) Calculate the decomposition efficiently by "building up" wedgelet projections at coarse scales from projections at finer scales.

(b) Impose geometrical constraints on how we expect contours in the image to behave by modeling the scale-to-scale dependence of the orientations of wedgelets used in the representation.

(c) Implement fast algorithms that find the wedgelet representation of an image that has the optimal trade-off between approximation error, sparsity, and geometrical faithfulness.

In addition, work presented in [4] shows these modeling techniques can be incorporated into practical image coders.

6. REFERENCES

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

[2] M. N. Do, P. L. Dragotti, R. Shukla, and M. Vetterli, "On the compression of two-dimensional piecewise smooth functions," in *IEEE Int. Conf. on Image Proc. – ICIP '01*, Thessaloniki, Greece, Oct. 2001.

[3] D. L. Donoho and X. Huo, "Beamlets and multiscale image analysis," in *Multiscale and Multiresolution Methods*, Lecture Notes in Computational Science and Engineeing. Springer, 2001.

[4] M. B. Wakin, J. K. Romberg, H. Choi, and R. G. Baraniuk, "Image compression using an efficient edge cartoon + texture model," in *IEEE Data Compression Conference, DCC*, Snowbird, Utah, April 2002, To Appear.

[4]The space of lines can be parameterized by the points on a cylinder embedded in $\mathbb{R}^3$, which is isomorphic to $\mathbb{R} \times [0, 2\pi)$; the distance between two lines is just the Euclidean distance between their representative points in $\mathbb{R} \times [0, 2\pi)$