MULTISCALE IMAGE PROCESSING USING NORMAL TRIANGULATED MESHES

Maarten Jansen, Hyeokho Choi, Sridhar Lavu, Richard Baraniuk

Dept. of Electrical and Computer Engineering
Rice University
Houston, TX 77005, USA

ABSTRACT

Multiresolution triangulation meshes are widely used in computer graphics for 3-d modeling of shapes. We propose an image representation and processing framework using a multiscale triangulation of the grayscale function. Triangles have the potential of *approximating* edges better than the blocky structures of tensor-product wavelets. Among the many possible triangulation schemes, normal meshes are natural for efficiently representing singularities in image data thanks to their *adaptivity* to the smoothness of the modeled image. Our non-linear, multiscale image decomposition algorithm, based on this subdivision scheme, takes edges into account in a way that is closely related to wedgelets and curvelets. The highly adaptive property of the normal mesh construction provides a very efficient representation of images, which potentially outperforms standard wavelet transforms. We demonstrate the approximation performance of the normal mesh representation through mathematical analyses for simple functions and simulations for real images.

1. INTRODUCTION

Images are typically acquired on a square grid of pixels, and so image processing algorithms typically operate on this square grid. Standard examples include filtering via convolution and the multiscale wavelet transform that builds a multiscale image representation corresponding to a range of different square pixel sizes. The square pixel grid is nearly always assumed to be fixed, with the dependent variable of the image the pixel intensity.

Although the acquisition and processing of image data on a square grid of pixels is simple, it turns out to be very inefficient for representing many important features in images, such as edges. Edges are the dominating features in real world images. Edges contain two types of information: *where* the edge is located and *what* is the step value (the height of the discontinuity). In 2-d functions (such as images), *where* information plays a crucial role, much more than in 1-d. In 2-d, edge singularities lie along 1-d contours. Although the dyadic wavelet transform is optimal for representing 1-d singularities, tensor-product wavelet transforms based on square pixels are ill-prepared to represent edges, since many wavelets overlap with the edge, leading to a preponderance of *where* information. Complicated new bases such as wedgelets and curvelets [1, 2] have been proposed as better alternatives for representing 2-d edge contours.

Supported by NSF grants CCR–99–73188, ONR grant N00014–99–1–0813, DARPA, and Texas Instruments. Contact info: {maartenj,choi,lavu,richb}@rice.edu — web: dsp.rice.edu

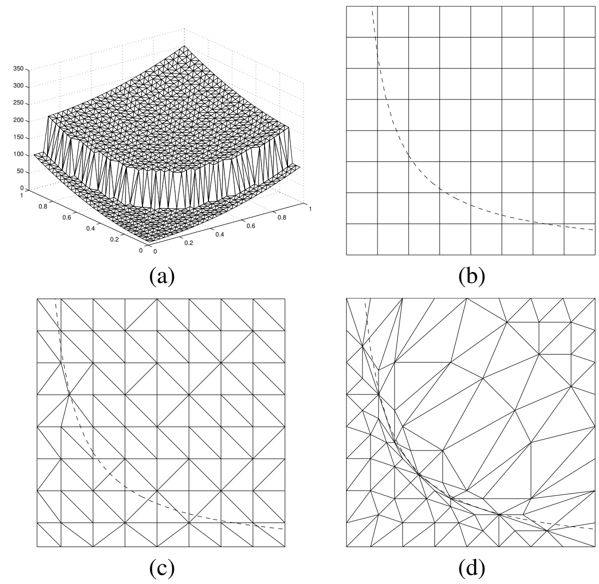

Fig. 1. Representing the function $f(x, y) = x^2 + y^2 + I_{\{y>0.1/x\}}$. (a) 3-d-plot of the function. (b) A square pixel based representation (such as that used by a tensor-product wavelet transform). The dashed line is the hyperbolic edge in $f(x, y)$. (c) A non-adaptive refinement has the potential of a better approximation, but it does not exploit this potential. (d) The combination of adaptivity and triangulation does the job.

An alternative to complicated bases over square grids for efficient edge contour representation is to treat images as special cases of 3-d surfaces and represent them using *triangular* image patches [3–5]. A triangulation consists of *triangles*, i.e., triplets of *vertices*, connected by *edges* (not to be confused with image edges). Because the edges of triangles can be placed in arbitrary locations and orientations, triangles have the potential to represent arbitrary edge contours (the *where* information) more accurately with a fewer number of patches than a fixed square-pixel representation. The key is to use an *adaptive triangulation* that places vertices more densely in edge regions for accurate and efficient edge representation, yielding a parsimonious image representation.

For efficient processing of mesh data, *multiscale* triangulation based on nonlinear subdivision has been proposed in computer graphics [6]. In this paper, we propose to use this multiscale mesh representation for images.

Multiscale mesh construction starts from a set of small num-

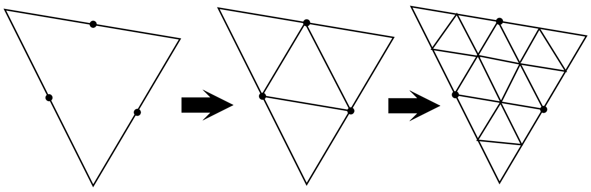

Fig. 2. The triangular subdivision principle: four triangles at finer scale are generated by subdividing each triangle edge at coarse scale.

ber of points on the surface. Finer triangular meshes are formed by subdivision, i.e., by gradually adding more data points (vertices, pixels). Unlike the subdivision scheme that places the new vertices corresponding to the midpoints of the triangle edges, we can adapt the location of the new vertices based on local geometry information. The *normal mesh* scheme selects the new points based on the local normal direction computed from the previous coarser scale mesh.

Originally developed for efficient 3-d surface representation, the normal mesh representation of an image shows remarkable adaptivity to the edge structure in the image. Starting with arbitrary set of initial vertices, we demonstrate that the normal mesh subdivision algorithm rapidly places more and more vertices directly on the edge, enabling direct representation of the *where* information of edges as well as the *what* information. Adaptivity and better approximation with triangles are the keys to the success of normal meshes as compared to wavelets.

In Section 2 we overview background on triangulations for general 3-d surfaces and images. Section 3 presents the normal mesh concept; section 4 specializes the construction to images. In Section 5 we demonstrate the superiority of multiresolution normal meshes over fixed tensor-product wavelet representations for images dominated by edges. Section 6 presents some practical results on synthetic and real images. Section 7 offers a discussion and conclusions.

2. MULTISCALE IMAGE TRIANGULATIONS

Consider the construction of a multiscale triangulated version of a square-pixel image using the principle of subdivision (vertex refinement). In standard subdivision for two-dimensional functions, we introduce vertices for the next finer scale at the midpoint of the existing triangle legs. This results in four child triangles that, in the function domain, cover the same area as their parent (see Fig. 2).

Once we start and fix the rules for refinement, we need only specify the initial vertex points. Finer scale vertices are uniquely defined without extra information. This is good for applications such as compression, where a parsimonious image representation is required.

We can now build a wavelet transform for this triangulated image representation. Because of the structure of the subdivision scheme, given the finest mesh, we can separate the points into two groups: those from the previous coarser scale (group A) and others obtained by the subdivision of these points (group B). With this "decimation" of points, we can apply the lifting scheme [7] to implement a wavelet transform on the triangular grid.

For each point in group B, we define a proper neighborhood window around it, and we predict the value at that point using the points of group A in the neighborhood (prediction step in lifting). The simplest algorithm merely applies simple linear prediction [7]. By subtracting the predicted value from the actual value, we obtain the wavelet coefficient. The scaling coefficients are obtained by updating the points in group A by adding the update values computed from the wavelet coefficients.

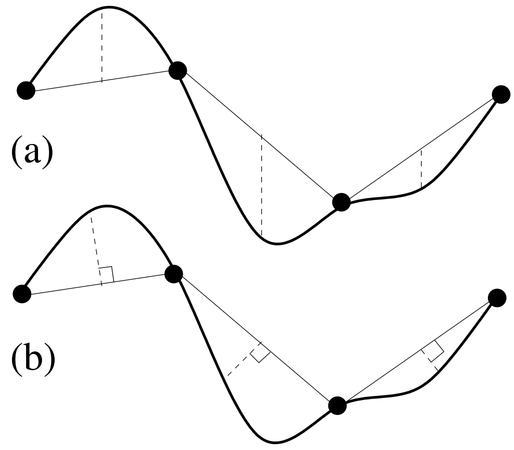

Fig. 3. (top) Computation of detail (wavelet) coefficient in the classical lifting scheme. (bottom) Normal mesh scheme.

3. NORMAL SUBDIVISION

The salient concepts of normal subdivision are easily described with a simple 1-d example. In a standard 1-d wavelet transform, a wavelet coefficient is computed as the vertical offset between a sample value and its prediction based on the neighboring samples. The length of the dashed, vertical line in Fig. 3 (top) is such a detail or wavelet coefficient: it tells how far the function value in the middle point deviates from a linear interpolation between two neighbors at coarser scale. Instead of linear interpolation, more sophisticated predictions could be used and more coarse scale neighbors could be involved.

The normal subdivision in Fig. 3(bottom) is very similar, except that it computes its offset in a direction *normal* to the current, coarse scale prediction. Thus, the detail coefficient tells us how far to go in a specific *direction*, supplying *where* information. This normal direction obviously depends on the coarse scale prediction, which makes the procedure non-linear.

4. NORMAL MESHES AND IMAGES WITH EDGES

Because an image is a function defined over the 2-d plane, we can treat an image as a grayscale surface in 3-d space and build a normal mesh for it. For example, a step edge in the image becomes a vertical surface in the 3-d space.

The basic steps to building a normal mesh of an image surface are as follows:

1. Choose the initial, coarsest scale vertices and connect them into a triangular mesh. In the simplest case, the initial vertices can be the set of four corner points of the image.

2. Repeat the following steps until all pixel points have been selected as triangle vertices:
   (a) For all edges in the current triangulation: compute the midpoint and the direction of the normal to the surface fitting a surface to the immediate current scale neighbors.
   (b) Find the point on the image surface that is pierced by the vector in the normal direction.
   (c) Record the displacement between the edge midpoint and the piercing point as the detail coefficient for the corresponding vertex.
   (d) Using the new vertices obtained, retriangulate each triangle to obtain the finer mesh through subdivision.

In some pathological cases, we need to go back to the usual midpoint subdivision scheme [6].

As illustrated in Fig. 1(d), the procedure to build a normal mesh has remarkable adaptivity to the locations of the edges in the image. With even an arbitrary choice of initial vertices, the normal mesh generation algorithm almost immediately starts placing the new vertices close to the edge, making the edges of the triangles align with the edge contours. As the triangles refine, the normal mesh provides a successive piecewise linear approximation of the edge contours, yielding very efficient representation of 2-d singularities with a small number of normal mesh detail coefficients.

The mechanism behind this interesting behavior is explained by the example of a simple 1-d step function, as illustrated in Fig. 4. The normal direction tends to show the way towards the singularities; if two points at coarse scale lie on both sides of an edge, then the normal piercing point is always closer to that edge than the initial subdivision point in the middle of the two coarse scale points. And once the intersample distance h becomes smaller than the edge height Δ, then the newly inserted point is always on the edge itself.

5. APPROXIMATION RATE FOR THE HORIZON CLASS

To see how well normal meshes approximate the contour singularities in images, we can analyze the adaptivity of the scheme for so-called Horizon Class Images [1], i.e. functions of the form:

$$f(x,y) = 1_{\{y>H(x)\}},$$

where we take for the horizon $H(x)$ a C^2-smooth function, as in Figure 1. The notation 1_A means that $f(x,y)$ is the indicator function on A: $f(x,y)=1$ if $(x,y)\in A$ and 0 otherwise. In 1-d, this horizon corresponds to a step function.

In 1-d, the normal mesh approach converges faster than wavelets on the step edge, because the first point on the vertical section of the curve breaks the approximation into two parts: from that moment the algorithm reduces to fitting to rectangular corners. We can prove that the average convergence rate over all starting points is:

$$\overline{\|\epsilon_n\|_2} = \mathcal{O}\left(2^{-\gamma n} i\right) \text{ with } \gamma \approx 1.26.$$

In this expression, $\epsilon_n = f - f_n$ is the approximation error function for a reconstruction from n detail coefficients. The actual convergence rate depends on the relative position of the edge with respect to the initial subdivision points, which is the reason for taking the average. We conjecture that the minimax rate is only a bit slower

$$\|\epsilon_n\|_{2,\text{minimax}} = \mathcal{O}\left(2^{-\frac{3}{4}\log_2 3n}\right),$$

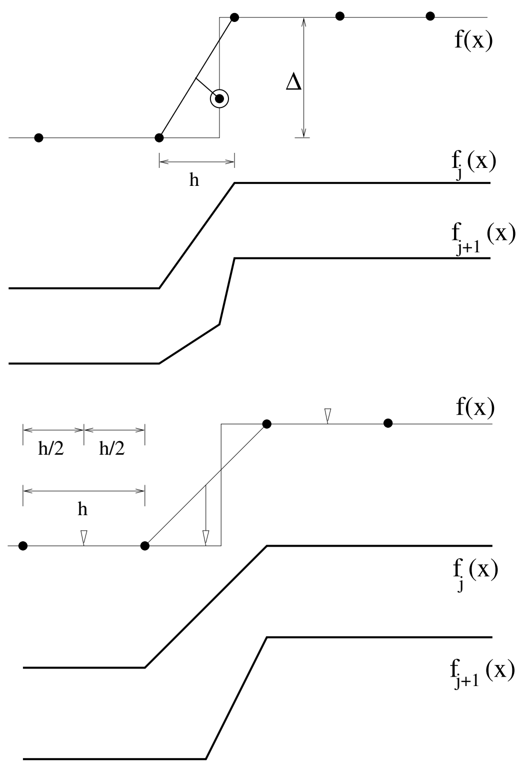

Fig. 4. The normal mesh approach at discontinuities (top), compared to the classical subdivision scheme (bottom).

and from a first analysis, a trivial lower bound for this minimax behavior is $\|\epsilon_n\|_{2,\text{minimax}} \leq \mathcal{O}\left(2^{-3/4n}\right)$, which is still faster than the wavelet rate $\|\epsilon_n\|_{2,\text{minimax}} = \mathcal{O}(2^{-n/2})$. Note that in 1-d, the step does not play any role if it is embedded in a piecewise smooth function: the general convergence rate for wavelets is $\|\epsilon_n\|_2 = \mathcal{O}(n^{-\alpha})$, α being the Lipschitz constant of the smooth part between the singularities (or the number of vanishing moments of the wavelet basis, if this is lower). The fact that a finite number of singularities does not affect the convergence rate is precisely the success of wavelets, and we expect the same behavior for 1-d normal meshes.

Although usual dyadic wavelet transforms show good approximation property for 1-d singularities, tensor-product 2-d wavelets are poor tools for approximating 2-d singularities along continuous contours. Unlike the 1-d case, the number of wavelet basis functions to cover the entire edge contour on the 2-d plane increases at a rate $\mathcal{O}(2^j)$ with scale j, cutting the approximation power of the 2-d wavelets (see Fig. 1).

For normal meshes, the picture becomes more interesting for 2-d contoured edges (see Fig. 1). While a non-linear wavelet approximation with n coefficients has an error of $\overline{\|\epsilon_n\|_2} = \mathcal{O}(n^{-1/2})$, we conjecture a rate of $\overline{\|\epsilon_n\|_2} = \mathcal{O}(n^{-1})$ for the normal mesh approach. The secret lies in the combination of triangulation and adaptivity. Triangles are able to trace the edge contour as piecewise linear, whereas the blocky structure of tensor product wavelets behaves like a piecewise constant approximation.

The error of a normal mesh approximation in 2-d is completely dominated by the error of this piecewise linear approximation of the *where* information in the edge. This observation suggests that "curved" triangles have the potential of catching the *where* information even better. The triangulation, as it is now, combines two objectives: it describes connectivity and at the same time it is used in the prediction of new points. There is no reason why the second aspect should be linked to the first one. Predicting new sample points on the edge could involve more than just two coarse scale points.

6. EXAMPLE AND RESULTS

Figure 5 compares the approximation power of a normal mesh decomposition with that of the classical CDF 2,2 tensor product wavelet basis. Both reconstructions are based on the 70 largest coefficients in the decomposition. The picture clearly illustrates the edge-adaptivity of the normal mesh approach.

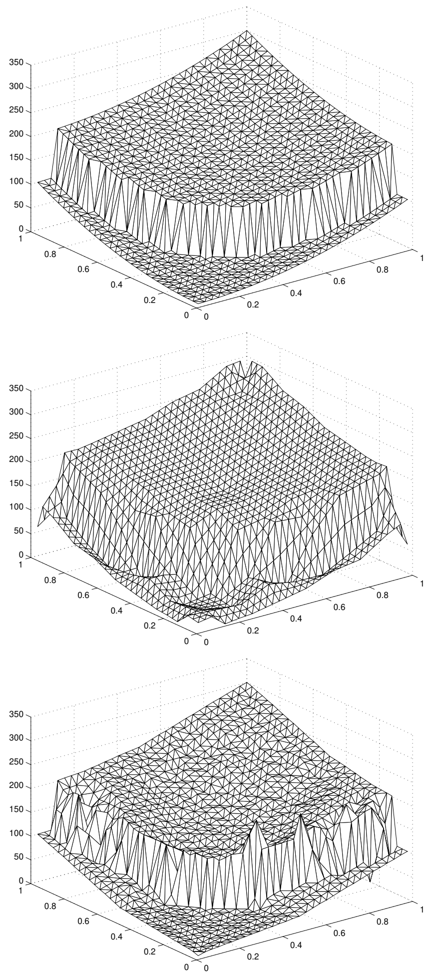

Fig. 5. (top) Original function. See Fig. 1(a). (middle) Tensor wavelet approximation using 70 out of 1024 wavelet coefficients. (bottom) Normal mesh approximation using 70 out of 1024 detail coefficients. The normal mesh approach clearly outperforms the wavelet approximation at edge preservation.

7. CONCLUSIONS

We have introduced an adaptive multiscale triangulation scheme for images based on a normal mesh decomposition. The scheme outperforms wavelet approximation thanks to the combined efforts of:

- adaptivity of the normal mesh approach. Normal mesh coefficients carry both *what* and *where* information about the edge.
- better approximation of edge contours when using triangulations instead of blocky tensor product wavelets.

Normal meshes can be used for image modeling, compression, and processing. However, algorithms will have to take into account that the decomposition is highly nonlinear.

8. REFERENCES

[1] D. L. Donoho, "Wedgelets: Nearly minimax estimation of edges," *Annals of Statistics*, vol. 27, no. 3, pp. 859–897, 1999.

[2] E. J. Candes and D. L. Donoho, "Curvelets — A surprisingly effective nonadaptive representation for objects with edges," Tech. Rep., Department of Statistics, Stanford University, 2000.

[3] S. G. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, CA, USA, 1998.

[4] S. Rippa, "Long and thin triangles can be good for linear interpolation," *SIAM J. Numer. Anal.*, vol. 29, no. 1, pp. 257–270, 1992.

[5] N. Dyn and S. Rippa, "Data-dependent triangulations for scattered data interpolation and finite-element approximation," *Appl. Num. Math.*, vol. 12, pp. 89–105, 1993.

[6] I. Guskov, K. Vidimee, W. Sweldens, and P. Schröder, "Normal meshes," in *SIGGRAPH 2000 Conference Proceedings*, 2000.

[7] W. Sweldens and P. Schröder, "Building your own wavelets at home," in *Wavelets in Computer Graphics*. 1996, ACM SIGGRAPH Course Notes, pp. 15–87, ACM.