

**Multiscale Approximation of Piecewise
Smooth Two-Dimensional Functions
using Normal Triangulated Meshes**

*Maarten Jansen
Richard Baraniuk
Sridhar Lavu*

Report TW 354, February 2003



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Multiscale Approximation of Piecewise Smooth Two-Dimensional Functions using Normal Triangulated Meshes

Maarten Jansen

Richard Baraniuk

Sridhar Lavu

Report TW 354, February 2003

Department of Computer Science, K.U.Leuven

Abstract

Multiresolution triangulation meshes are widely used in computer graphics for representing three-dimensional (3-d) shapes. We propose to use these tools to represent 2-d piecewise smooth functions such as grayscale images, because triangles have potential to more efficiently approximate the discontinuities between the smooth pieces than other standard tools like wavelets. We show that *normal mesh subdivision* is an efficient triangulation, thanks to its local *adaptivity* to the discontinuities. Indeed, for the so-called *horizon class* of functions comprising constant regions separated by smooth discontinuities, we show that the normal mesh representation has an optimal error decay rate as the number of terms in the representation grows. This optimal decay rate is possible because normal meshes automatically generate a polyline (piecewise linear) approximation of each discontinuity, unlike the blocky piecewise constant approximation of tensor product wavelets. In this way, the nonlinear multiscale normal mesh representation is closely related to the recently developed wedgelet and curvelet transforms.

Keywords : normal, offsets, mesh, image, multiresolution, wavelet, approximation

AMS(MOS) Classification : 41A46, 41A63, 42C40.

Multiscale Approximation of Piecewise Smooth Two-Dimensional Functions using Normal Triangulated Meshes

Maarten Jansen^{◊*}, Richard Baraniuk,[◊] and Sridhar Lavu^{◊†}

[◊] Department of Mathematics and Computer Science
Technical University of Eindhoven
PO Box 513 NL-5600 MB, Eindhoven, The Netherlands
and
Department of Computer Science
K.U.Leuven, Belgium

[◊] Department of Electrical and Computer Engineering
Rice University
6100 Main Street, Houston, TX 77005, USA

February 10, 2003

Abstract

Multiresolution triangulation meshes are widely used in computer graphics for representing three-dimensional (3-d) shapes. We propose to use these tools to represent 2-d piecewise smooth functions such as grayscale images, because triangles have potential to more efficiently approximate the discontinuities between the smooth pieces than other standard tools like wavelets. We show that *normal mesh subdivision* is an efficient triangulation, thanks to its local *adaptivity* to the discontinuities. Indeed, for the so-called *horizon class* of functions comprising constant regions separated by smooth discontinuities, we show that the normal mesh representation has an optimal error decay rate as the number of terms in the representation grows. This optimal decay rate is possible because normal meshes automatically generate a polyline (piecewise linear) approximation of each discontinuity, unlike the blocky piecewise constant approximation of tensor product wavelets. In this way, the nonlinear multiscale normal mesh representation is closely related to the recently developed wedgelet and curvelet transforms.

1 Introduction

1.1 Representations of piecewise smooth function

This paper concerns the representation and approximation of piecewise smooth, two-dimensional (2-d) functions, which consist of smooth regions delineated by step discontinuities along smooth 1-d contours, which

*Maarten Jansen is a postdoctoral fellow with the Fund of Scientific Research (FWO), Flanders (Belgium)

†Supported by NSF, ONR, AFOSR, DARPA, and the Texas Instruments Leadership University Program.
Email: mjansen@win.tue.nl, richb@rice.edu, lavu@rice.edu; Web: www.cs.kuleuven.ac.be, dsp.rice.edu

we will call *edges*. Many different types of real-world data can be modeled as piecewise smooth. As an important example, a piecewise smooth function is a quite accurate model for a grayscale *image*, which represents the light intensity of a black-and-white visual scene. While we will use images as our central, running example in this paper, other examples abound in statistics and differential equations for a broad spectrum of applications.

By *approximation*, we mean approximating a piecewise smooth function with a finite dimensional representation. Immediate applications of approximation results include compression and noise removal (denoising).

For images and many other kinds of data, an approximation is typically defined on a discrete set of points on some grid. For example, digital images are typically acquired by sampling the light intensity at discrete points on a square grid of pixels (currently using a CCD array), and so image representations and processing algorithms typically operate on this square grid. The square pixel grid is nearly always assumed to be fixed, with the dependent variable of the image the pixel intensity. While 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 image features, including the *edges*.

Edges are the dominating features in piecewise smooth 2-d functions. 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, *where* information plays a crucial role, much more than in 1-d. In 1-d piecewise smooth functions, discontinuities occur at isolated points, and these can be easily captured in a wavelet transform. In 2-d, edge singularities lie along 1-d contours, which are much harder to capture.

1.2 Wavelets

The time-scale analysis of the wavelet representation provides a powerful tool for approximating a 1-d function f . Thanks to the local support of the basis functions, under mild conditions, a nonlinear wavelet approximation f_n containing the n largest terms of the wavelet expansion of f performs as well on a piecewise smooth f as on a smooth f [14, 15, 7, 6, 10]. Indeed, the L_2 approximation error decays rapidly with increasing n :

$$\|f - f_n^{1\text{-d wavelet}}\| = \mathcal{O}(n^{-\nu}). \quad (1)$$

In this equation, ν stands for

$$\nu = \min(\tilde{p}, \alpha),$$

with \tilde{p} the number of (dual) vanishing moments of the wavelet analysis and α the Lipschitz regularity of the signal at its non-singular points. Wavelets provide a very efficient representation of 1-d piecewise smooth signals primarily because in 1-d the *where* information consists of merely a few isolated points.

Wavelets are thus well-suited for estimating a piecewise smooth 1-d function in the presence of noise. In the minimax sense, the performance of a simple n -term approximation algorithm comes within a neglectible logarithmic factor of the best possible method involving a piecewise polynomial with knots at the (assumed known) positions of the singularities [9].

Unfortunately, this approximation power does not carry over into two and higher dimensions. Indeed, standard tensor-product wavelet transforms based on a square grid of 2-d sampling points are ill-prepared to represent edges, since many wavelets overlap with the 1-d edge, leading to a preponderance of *where* information (see Fig. 1).

Given a 2-d function f that is smooth except for an edge singularity along a smooth (say C^2) curve, the nonlinear wavelet approximation f_n using the n largest wavelet terms has an L_2 error rate

$$\|f - f_n^{2\text{-d wavelet}}\| = \mathcal{O}(n^{-1/2}).$$

This outperforms a Fourier procedure, where about the best we can do is a linear approximation taking the *first* n Fourier coefficients

$$\|f - f_n^{2\text{-d Fourier}}\| = \mathcal{O}(n^{-1/4}).$$

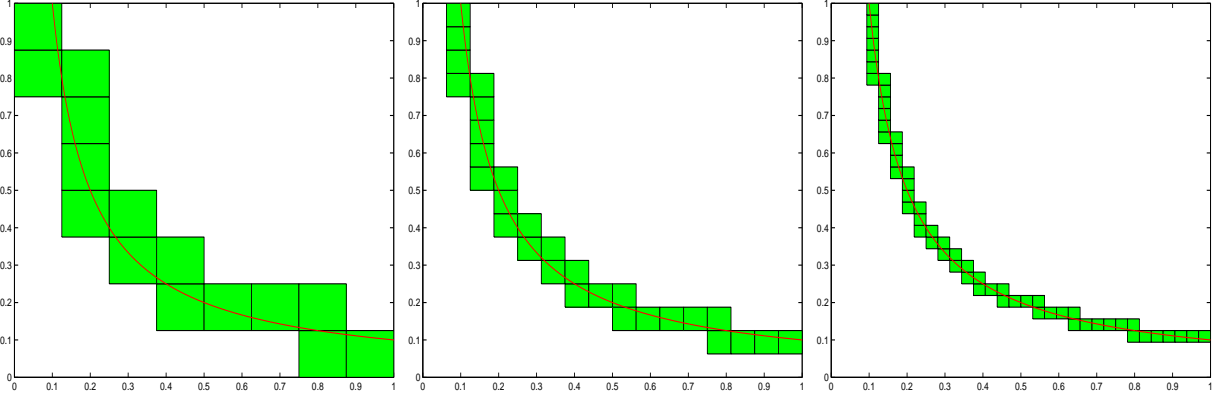


Figure 1: *Haar wavelet approximation of a 2-d piecewise constant image featuring a smooth 1-d edge singularity. Each shaded square corresponds to the support of a wavelet basis function. At each finer scale, an increasing number of wavelets is necessary to cover (and hence represent) the singularity. This effect does not exist in 1-d and explains the suboptimal performance of wavelets for representing 2-d piecewise smooth functions. Another drawback of tensor product wavelets is that they approximate the edge curve as a piecewise constant. This explains the “blockiness” of wavelet image approximations.*

Nevertheless, neither of these procedures comes close to the 1-d rate of (1). This is partly due to an inherent dimensionality effect: approximation of 2-d data is inevitably more difficult than 1-d data. Yet, wavelets do not obtain the optimal 2-d rate either. They approximate a curved singularity as a piecewise constant. This observation explains the blocky output of wavelet image approximations.

In order to achieve better approximation rates on 2-d edge contours, complicated new bases such as wedgelets and curvelets [8, 1] have been proposed as better alternatives for representing 2-d edge contours. Treatment of line singularities is an important subject of ongoing research [8, 1, 19, 2, 3]. Most of these alternative decompositions are overcomplete, and some involve complicated computational algorithms to form an approximation.

1.3 Triangular meshes

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 [15, 17, 12]. A triangulation consists of *triangles*, that is, triplets of *vertices* connected by *edges* (not to be confused with image edges). Because the triangles edges 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 grid 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 (see Fig. 2).

Indeed, an adaptive triangle-based decomposition can provide a piecewise linear edge approximation, provided that the triangulation adapts itself to the precise locations of the edges. Ideally, this could lead to an error rate of

$$\|f - f_n^{2-d \text{ opt. triangle}}\| = \mathcal{O}(n^{-1}), \quad (2)$$

For efficient processing of 3-d mesh data, *multiscale* triangulation based on nonlinear *subdivision* has been proposed in computer graphics. Multiscale mesh construction starts from a small number of coarse-scale points on the surface. Finer triangular meshes are formed by subdividing, that is, by gradually adding more data points (vertices, pixels). Unlike the standard subdivision scheme that places new vertices at the midpoints of the triangle edges, we can adapt the location of the new vertices based on local geometry

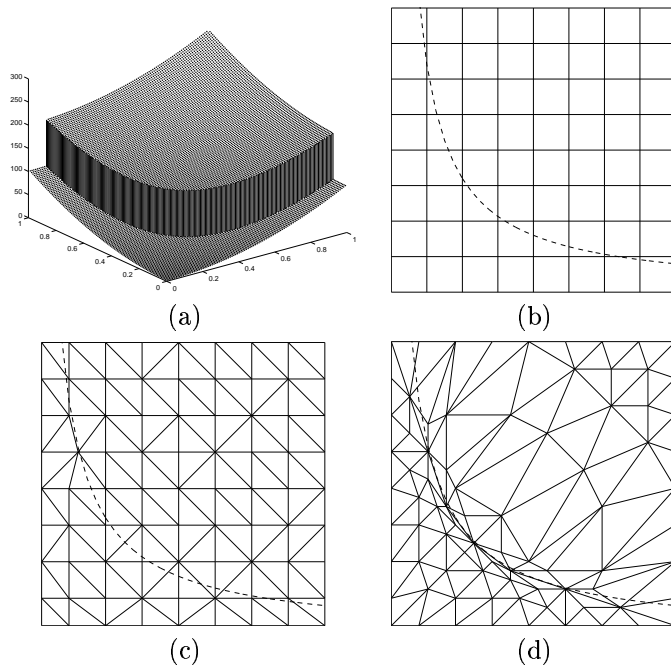


Figure 2: Representing the 2-d function $f(x, y) = x^2 + y^2 + I_{\{y > 0.1/x\}}$. (a) 3-d mesh plot of the function. (b) A square grid representation such as that used by a tensor-product wavelet transform. The dashed line is the hyperbolic edge in $f(x, y)$. (c) A non-adaptive triangular refinement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

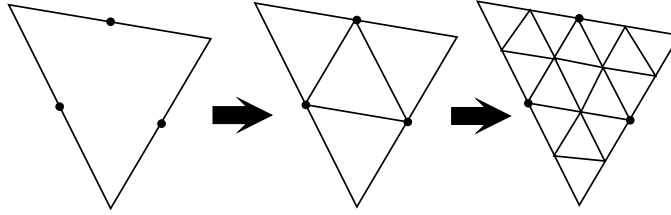


Figure 3: *The triangular subdivision principle: four triangles at a finer scale are generated by subdividing each triangle at the next coarser scale.*

information. The *normal mesh* scheme selects the new points based on the local normal direction computed from the previous coarser scale mesh [13].

Originally developed for efficient 3-d surface representation, we will show that the normal mesh representation shows remarkable adaptivity to the edge structure of 2-d piecewise smooth functions (see Figure 2(c)). Starting with an arbitrary set of initial vertices, we will demonstrate that the normal mesh subdivision algorithm rapidly places more and more vertices directly on the edge contour, enabling a 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 opposed to wavelets.

In this paper, we propose a multiscale normal mesh representation for piecewise smooth 2-d functions such as images. We will show that for the idealized *horizon class* of images that are piecewise constant save for a C^2 edge discontinuity we obtain the optimal error decay rate of (2).

The idea of data-adaptive triangulation has been elaborated under different assumptions in [11]. In this paper, the triangulation is constructed based on an *existing* set of vertices. These vertices coincide with samples of the function and remain fixed. The normal offset method on the other hand first finds the best locations for the vertices before triangulating them. These locations are chosen for optimal approximation.

1.4 Paper overview

In Section 2 we overview background on triangulations for general 3-d surfaces, present the normal mesh concept, and specialize the construction to 2-d piecewise smooth functions. The rest of the paper conducts a detailed performance analysis for horizon class images in two steps. In Step I (Sections 3 and 4), we analyze the normal “mesh” (polyline) approximation in 1-d. In Step II (Section 6), we leverage this analysis into the 2-d horizon class case. Section 7 presents some practical results on synthetic and real images. Section 8 offers a discussion and conclusions.

2 Multiscale Image Triangulations

2.1 Quadtree triangulations

Consider the construction of a multiscale triangulated function representation using the principle of subdivision (vertex refinement). In standard subdivision for 2-d 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. 3).

Once we begin 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 valuable in applications such as compression, where a parsimonious image representation is required.

To build a wavelet transform for a multiscale triangulated image representation, we separate the vertices

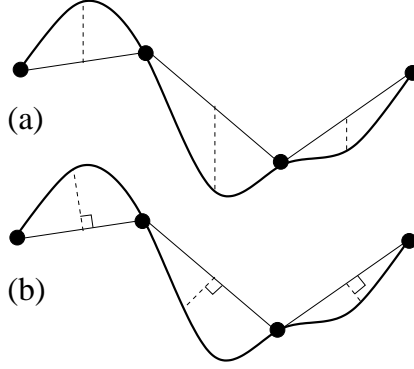


Figure 4: *1-d wavelet coefficient computation in (a) the classical lifting scheme and (vertical offsets) (b) the normal mesh scheme (normal offsets).*

from the finest scale mesh into two groups: those from the previous coarser scale (group A) and those obtained by the subdivision of these points (group B). With this “decimation” of points, we can apply the lifting scheme [18] 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 from group A in the neighborhood. The simplest algorithm merely applies simple linear prediction [18]. 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 update values computed from the wavelet coefficients.

2.2 Normal subdivision in 1-d

The salient concepts of normal subdivision are easily described in 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 its neighboring samples. The length of the dashed, vertical line in Fig. 4(a) is such a wavelet coefficient; it tells how far the function value in the middle point deviates from a linear interpolation of its two neighbors at the next 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. 4(b) 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, not just vertically, but in a specific *direction*, supplying *where* information. This normal direction obviously depends on the coarse scale prediction, which makes the procedure nonlinear.

2.3 Normal subdivision in 2-d

We can treat a 2-d piecewise smooth function as a surface in 3-d space and build a normal mesh for it. Step edges in the 2-d function now become vertical 3-d surfaces. The basic steps to building a normal mesh of a 2-d function run 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 function (assuming it has a finite domain).
2. Repeat the following steps for each edge of the triangulation until the approximation converges:

- (a) Compute the midpoint of the edge; this becomes a new vertex. Compute the direction of the vector normal to the surface that fitting the immediate neighbors of the edge.
- (b) Find the point on the function surface that is pierced by the normal vector. (In practice, with a sampled 2-d function, the surface is defined locally by fitting the given sample points.)
- (c) Record the displacement between the new vertex and the piercing point as the wavelet coefficient for the new vertex.
- (d) Using the new vertices, 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 [13].

As illustrated in Fig. 2(d), this procedure builds a normal mesh with remarkable adaptivity to the locations of the singularities in the function. Even with an arbitrary choice of initial vertices, the normal mesh algorithm almost immediately starts placing new vertices close to the singularities, making the edges of the triangles align with the function contours. As the triangles refine, the normal mesh provides a successive piecewise linear approximation of each contour, yielding an efficient representation of 2-d singularities with a small number of normal mesh detail coefficients.

The mechanism behind this interesting behavior is explained by considering a simple 1-d step function example, as illustrated in Fig. 5. *The normal direction tends to point to the singularities*; if two vertices at a coarse scale j lie on either side of a singularity, then the normal piercing point is always closer to the singularity than the standard midpoint subdivision point. And once the inter-vertex distance h becomes smaller than the singularity height Δ , then the newly inserted point is always on the singularity itself.

We will now analyze the performance of 2-d nonlinear approximation based on normal mesh triangulations for horizon class images. We proceed in two steps. Step I (Sections 3 and 4) analyzes the performance of normal “meshes” (polylines) at representing 1-d step discontinuities. Step II (Section 6) then extends this analysis to take into account the continuous curve of 1-d step discontinuities present in a horizon class image.

3 Analysis I: Normal Polylines for 1-d Piecewise Constant Functions

3.1 Rapid localization of singularity position

Suppose we have a step function $f(x) = \Delta \cdot 1_{\{x > x_0\}}$ with a discontinuity at $x_0 \in (0, 1)$ and call $\Delta = f(x_0+) - f(x_0-) = 1$ the *height* of the discontinuity. We wish to construct a multiscale approximation f_j of this function, where j denotes the scale (or resolution level). The normal mesh subdivision scheme generates an irregular grid \mathbf{x}_j near this singularity. In this analysis, we consider normal offsets with respect to a prediction by polylines, i.e., piecewise linear polynomials connecting . We call h_j the uncertainty about the location (the ‘where’ information) of the jump, i.e., the width of the sample interval $[x_{j,i}, x_{j,i+1}]$, containing the singularity x_0 : $h_j = x_{j,i+1} - x_{j,i}$ and $x_{j,i} \leq x_0 \leq x_{j,i+1}$. By construction in Figure 5 (top), it holds that $h_j \leq h_{j-1}/2$, so

$$h_j = \mathcal{O}(2^{-j}).$$

If h_j becomes smaller than Δ , then for certain $h_{j+1} = 0$, i.e., the next subdivision point lies on the singularity. This singularity locating property becomes crucial in 2-d, as we discuss later.

3.2 Expected behaviour near singularities

After having located the singularity position, the algorithm breaks down into two independent approximations: the behaviour left of the singularity is independent of, though similar to the behaviour on the right, so we study (the left) one side only, as in Figure 6. For further reference, we let the first subdivision point on the singularity correspond to level $j = 0$.

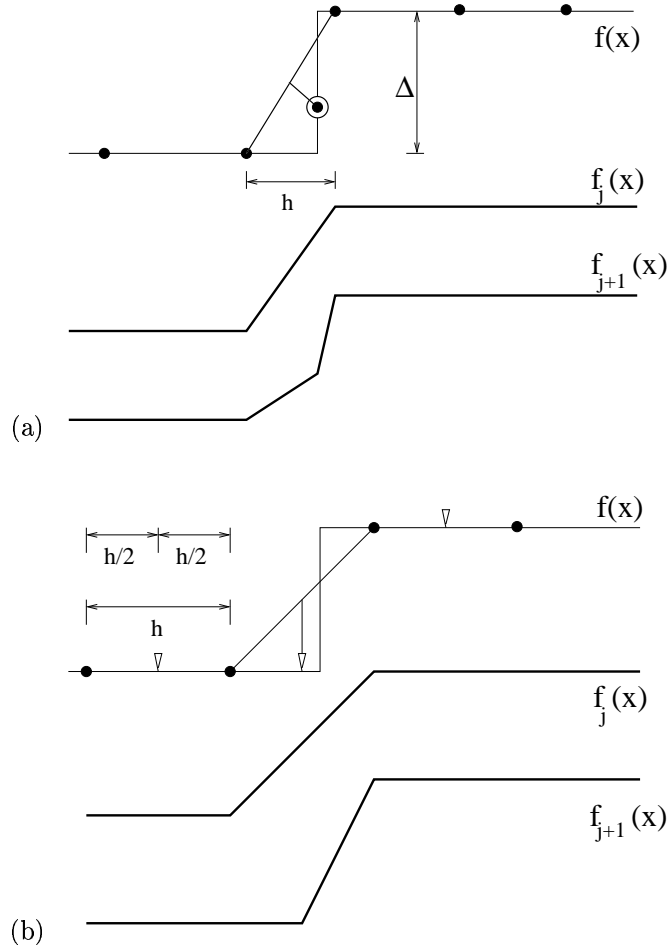


Figure 5: (a) Normal mesh approach at a discontinuity, compared to (b) the classical subdivision scheme. In the normal mesh case, the approximation f_{j+1} at scale $j + 1$ has a point on the exact location of the singularity, because the normal offset shows the way towards this singularity.

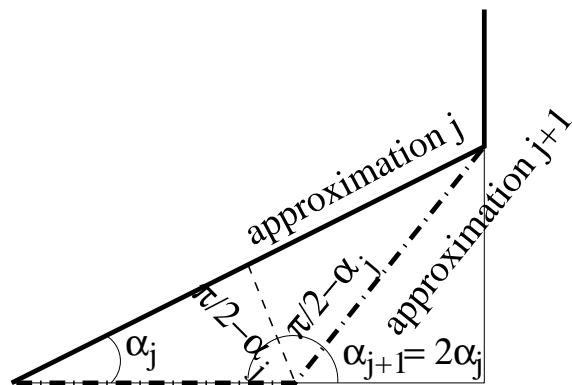


Figure 6: Evolution of the approximation near singularities throughout successive scales.

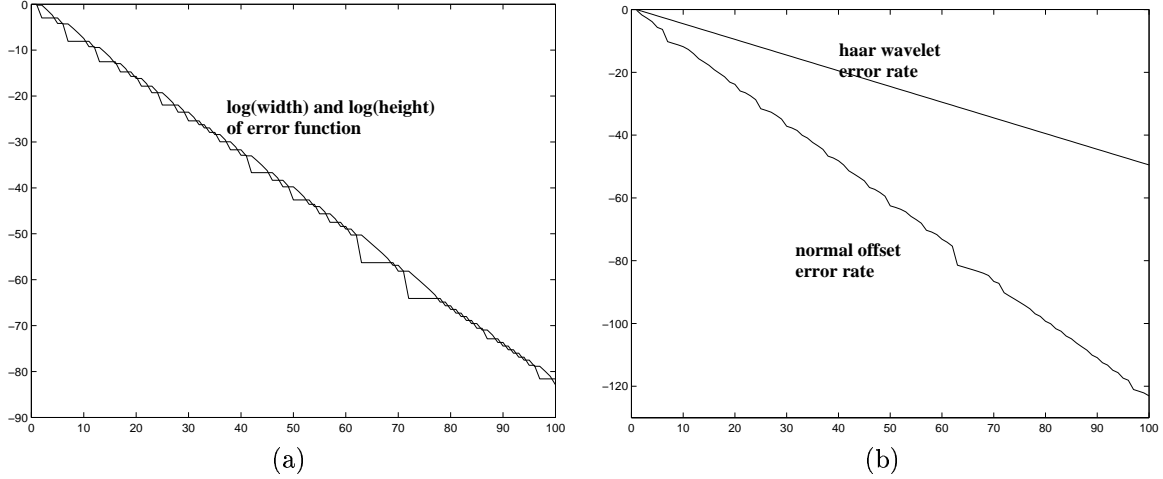


Figure 7: (a) Evolution of the logarithm of the error support and the error L_∞ -norm for a normal mesh approximation of a step function. (b) Log of the L_2 -norm of error function as function of n , the number of detail coefficients.

The procedure gradually reduces the error near the singularity, which essentially happens in two possible ways, depending on the slope of the present approximation, i.e., the angle α_j in the picture. If $\alpha_j < \pi/4$, as in the figure, the normal direction pierces the step function in its flat (horizontal) region, thereby reducing the interval on which the approximation differs from the true function. We call S_j the width of this interval: it is the support of the (left side of) the approximation error. On the other hand, if $\alpha_j > \pi/4$, the normal direction finds a new point on the singularity, and so it reduces the *height* H_j of the error function. H_j is also the L_∞ norm of the (left side) error function.

It holds that:

1. If $\alpha_j < \pi/4$ then

- $S_{j+1} = \frac{S_j}{2} (1 - \tan^2 \alpha_j) = S_j \frac{\tan \alpha_j}{\tan 2\alpha_j}$
- $H_{j+1} = H_j$
- $\alpha_{j+1} = 2\alpha_j$

2. If $\alpha_j > \pi/4$ then

- $S_{j+1} = S_j$
- $H_{j+1} = \frac{H_j}{2} (1 - \cot^2 \alpha_j)$
- $\alpha_{j+1} = 2\alpha_j - \frac{\pi}{2}$

In other words, if $\alpha_j > \pi/4$, the support width of the error function remains unchanged, but if it changes, i.e., if $\alpha_j < \pi/4$, it is reduced by a factor of more than a half. This means that the convergence is not monotone, and the precise process depends on the initial angle α_0 . Figure 7(a) plots $\log_2 S_n$ and $\log_2 H_n$ and Figure 7(b) compares the logarithmic error $\log_2 \|\epsilon_n\|_2 = \log_2 \sqrt{S_n H_n^2 / 3}$ with that of Haar wavelets ($\log_2 \|\epsilon_n\| = \mathcal{O}(-n/2)$). Note that the number of coefficients n equals the scale j . We could state all results in terms of n , but we chose to use j as subscript wherever a result or argument is based on thinking in scales.

It is interesting to analyse the behaviour of a wavelet approximation in this framework. Figure 8 shows that the support of the error function is divided by two in every step, but the height (i.e., the difference between maximum and minimum of the error curve) remains a constant. This leads to a L_2 -error rate of $\mathcal{O}(2^{-n/2})$.

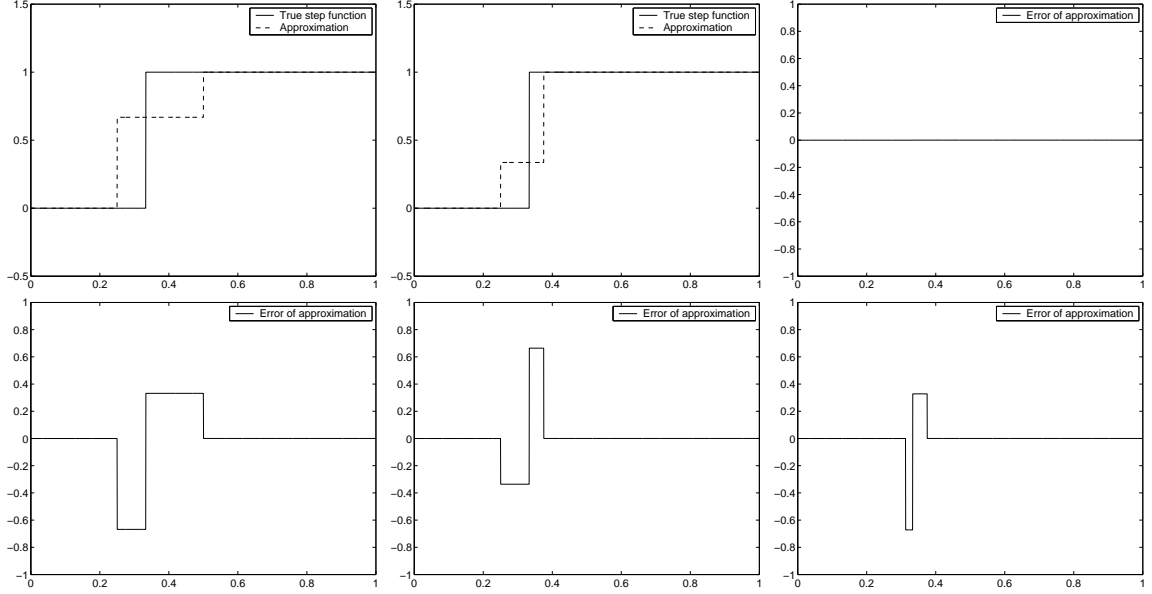


Figure 8: The evolution of the error of a Haar approximation for a step function on $[0, 1]$ with step at $1/3$. The 3 plots on top compare the approximation with the true function at successive scales. The plots below show the corresponding error function. The error support width is reduced by a factor 2 in every step, but the error height remains a constant.

Theorem 1 *If the initial angle α_0 is uniformly distributed on $[0, \pi/2]$, then all subsequent α_j are uniformly distributed, and the expected logarithmic error reduction is:*

$$\gamma := \mathbb{E} \log_2 \frac{\|\epsilon_{n+1}\|}{\|\epsilon_n\|} = -\frac{3}{2} - \frac{6}{\pi \log 2} \int_0^{\pi/4} \log \cos \alpha \, d\alpha$$

Proof:

$$\begin{aligned} \gamma &:= \mathbb{E} \log_2 \frac{\|\epsilon_{n+1}\|}{\|\epsilon_n\|} = \mathbb{E} \log_2 \sqrt{\frac{S_{n+1} H_{n+1}^2}{S_n H_n^2}} \\ &= \frac{1}{2} \frac{1}{\pi/2} \left(\int_0^{\pi/4} \log_2 \left(\frac{1 - \tan^2 \alpha}{2} \right) d\alpha + \int_{\pi/4}^{\pi/2} \log_2 \left(\frac{1 - \cot^2 \alpha}{2} \right)^2 d\alpha \right) \\ &= \frac{1}{\pi} \left(3 \int_0^{\pi/4} \log_2 \left(\frac{\cos^2 \alpha - \sin^2 \alpha}{\cos^2 \alpha} \right) d\alpha + 3 \int_0^{\pi/4} \log_2 \frac{1}{2} d\alpha \right) \\ &= \frac{3}{\pi} \left(-\frac{\pi}{4} \log_2 2 + \int_0^{\pi/4} \log_2 \frac{\cos 2\alpha}{\cos^2 \alpha} d\alpha \right) \\ &= \frac{3}{\pi} \left(-\frac{\pi}{4} + \frac{1}{2 \log 2} \int_0^{\pi/2} \log \cos \alpha \, d\alpha - \frac{2}{\log 2} \int_0^{\pi/4} \log \cos \alpha \, d\alpha \right) \\ &= \frac{3}{\pi} \left(-\frac{\pi}{4} - \frac{\pi}{4} - \frac{2}{\log 2} \int_0^{\pi/4} \log \cos \alpha \, d\alpha \right) \\ &\approx -\frac{3}{2} + 0.24 = -1.26 \end{aligned}$$

We used the fact that $\cot \alpha = \tan(\pi/2 - \alpha)$, to reduce all integrals to the interval $[0, \pi/4]$ and we filled in

the known integral

$$\int_0^{\pi/2} \log \cos \alpha \, d\alpha = \pi \log 2/2.$$

□

Note that we could make this analysis, because the uniform density is invariant under the refinement scheme for α_j . Starting from any other density with bounded derivative, the successive density functions rapidly converge to a constant:

Lemma 1 *If f_n is the density function of the angle α_n after n refinement steps, and if the initial density f_0 has a bounded derivative on $[0, \pi/2]$, then*

$$\lim_{n \rightarrow \infty} f_n(\alpha) = 2/\pi.$$

Proof: Call

$$A = \max_{\alpha \in [0, \pi/2]} |f_0'(\alpha)|,$$

and note that

$$f_n(\alpha) = \frac{1}{2}f_{n-1}\left(\frac{x}{2}\right) + \frac{1}{2}f_{n-1}\left(\frac{x}{2} + \frac{\pi}{4}\right).$$

It then follows immediately that all f_n are continuous and differentiable and that

$$|f_n'(\alpha)| \leq 2^{-n}A.$$

□

3.3 Ergodicity

The question arises whether the *average* error reduction per step, i.e., the mean reduction taken over a sequence of scales, converges to the expected error reduction in Theorem 1. Figure 7(a) seems to confirm this for an arbitrary initial α_0 .

It is interesting though to observe that

$$\alpha_j = \text{frac}\left(\frac{\alpha_0}{\pi/2}2^j\right)\frac{\pi}{2},$$

where $\text{frac}(x)$ is the fractional part of a real number x . If we call

$$\beta_j = \frac{\alpha_j}{\pi/2}, \tag{3}$$

then we see that in binary notation β_j is the fractional part of the number obtained by shifting the point in β_0 over j positions to the right. Then it is clear that

$$\beta_0 = .010011000111000011110000011111000000111111 \dots$$

induces a sequence with two accumulation points (0 and 1) and that the α_j do not fill up the interval $[0, \pi/2]$ uniformly. This holds even more for $\beta_0 \in \mathbb{Q}$, in which case only a limited subset of possible angles show up ever again. $\alpha_0 = \pi/4$ leads to immediate convergence. Another example is $\beta_0 = .01010101 \dots$ corresponding to $\alpha_0 = \pi/6$. In this case α_j alternates between $\alpha_{2j} = \pi/6$ and $\alpha_{2j+1} = \pi/3$, and it is easy to check that the convergence rate for the L_2 -norm is then $\log_2(3\sqrt{3})/2 \approx 1.19$, which is a bit slower than the average. The next section shows that this is the slowest possible convergence.

There is an immediate and interesting connection with the notion of “normal numbers” in number theory [16, Chapter8]. A normal number in base b is a number where any configuration of n digits (for any choice

of n) appears with frequency $1/b^n$. If β_0 is normal in the 2-base, this means that any sequence of n bits in its binary notation, appears with frequency $1/2^n$. The β_j following from the refinement scheme on a normal β_0 are uniformly distributed. Indeed, as mentioned before, the uniform density is invariant under the refinement map. In measure theory, it is said that the refinement $T(\beta) = \text{frac}(2\beta) := 2\beta - \lfloor 2\beta \rfloor$ on the unit interval with Lebesgue measure (i.e., uniform distribution) is an *endomorphism*. In this definition $\text{frac}(\cdot)$ denotes the fractional part of a real number and $\lfloor \cdot \rfloor$ stands for the integer part. It can be proven that this endomorphism is *ergodic*, i.e., $T^{-1}(A) = A$ is only possible if $P(A) = 0$ or $P(A) = 1$. Birkhoff's ergodic theorem then guarantees that the average of any (measurable) function $f(\beta)$ over the observed values $\beta_j = T(\beta_{j-1})$ converges and that the limit equals a.s. the expectation of this function with respect to (in our case) the Lebesgue measure, i.e., the uniform density. If we take for $f(\beta)$ the characteristic function on an interval $[0, B]$, for arbitrary $0 \leq B \leq 1$, we see that the frequency of $\beta_j \leq B$ equals B , which is the cumulative of the uniform density. This proves that almost all initialisations β_0 induce a uniformly distributed sequence. The set of such initial β_0 equals the set of normal numbers in $[0, 1]$.

3.4 Minimum convergence rate

Theorem 2 *The error ϵ_j of a normal polyline approximation of a singular point in a piecewise constant function as in Figure 6 satisfies:*

$$\bar{\gamma} := \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{j=1}^n \log_2 \frac{\|\epsilon_j\|_2}{\|\epsilon_{j-1}\|_2} \leq -\log_2(3\sqrt{3})/2 \approx -1.19.$$

From the analysis in Section 3.2, it is clear that the function $\bar{\gamma}(\alpha_0)$ shows a sort of fractal behaviour and simple optimisation techniques do not apply. In order to prove the theorem, we specifically show that $\alpha_0 = \pi/6$ leads to the slowest convergence rate. To do so, we start with some quick observations:

Lemma 2 *If the sequence α_j is generated as defined in Section 3.2, then $\forall j \in \mathbb{Z}$*

$$\bar{\gamma}(\alpha_j) = \bar{\gamma}(\alpha_0).$$

In particular: the asymptotic behaviour for $\alpha_0 = \beta_0\pi/2$ is the same as that for $\alpha_1 = \beta_1\pi/2$, where β_1 is the fraction obtained by shifting the binary representation of β_0 up to the first 0-1-switch.

So, in $.0000001\dots\pi/2$, the first zeros can be omitted for asymptotic analysis. This is a trivial asymptotic argument. It limits the search for the maximum value of $\bar{\gamma}(\alpha_0)$ to the interval $(\pi/8, 3\pi/8)$.

Lemma 3 *Measured in L_1 -norm, the error decay starting from initial angle α_0 is exactly the same as the error behaviour for $\pi/2 - \alpha_0$.*

This is clear from Figure 9. Unlike wavelets, the normal polyline approximation does not treat a signal as a function, and therefore it does not see jumps as discontinuities. Horizontal and vertical line segments are treated equally.

In L_1 , investigating β_0 is equivalent to investigating $1 - \beta_0$. In other words, switching all zeros and ones in β_0 's binary representation does not change the L_1 behaviour.

Lemma 4 *Minimising the asymptotic L_p decay is equivalent to minimising the asymptotic L_1 behaviour.*

This is because $\|\epsilon_n\|_p^p = S_n H_n^p$, and, once more, the normal polyline approximation does not make a difference between vertical and horizontal sections of a piecewise constant signal. As a consequence, $H_n = \mathcal{O}(S_n)$, from which the Lemma follows. Lemmata 3 and 4 further restrict the maximisation to the interval $(\pi/4, 3\pi/8)$.

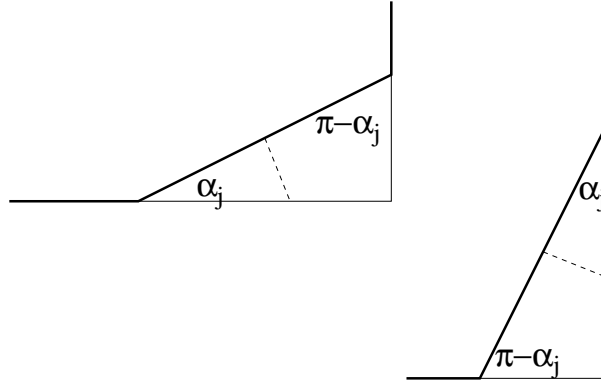


Figure 9: Normal polyline approximation near a singularity. The approximation is determined by the angle α_j , but α_j and $\pi - \alpha_j$ correspond to the same L_1 error.

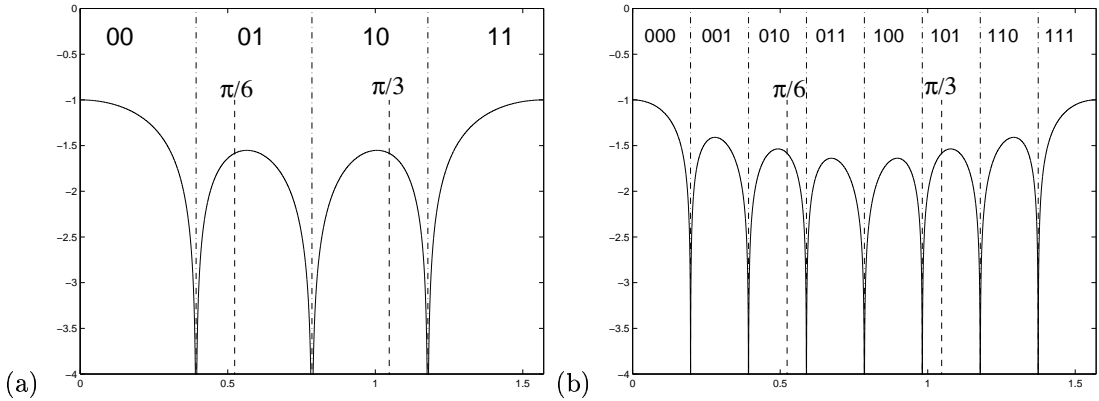


Figure 10: (a) Mean two steps log-average decay of the L_1 -error as a function of the initial angle α_0 . (b) Mean three steps log-average decay of the L_1 -error as a function of the initial angle α_0 . The binary figures correspond to the first two, resp. three digits in the binary representation of $\beta_0 = \alpha_0/(\pi/2)$.

Figure 10 plots the two and three steps log-average decay of the L_1 -error as a function of the initial angle α_0 . In general, we call

$$\bar{\eta}_n(\alpha_0) := \frac{1}{n} \sum_{j=1}^n \log_2 \frac{\|\epsilon_j\|_1}{\|\epsilon_{j-1}\|_1}$$

the n steps average decay of the L_1 -error. Note that $\eta_n(\pi/6) = -\log_2 3 = \eta_n(\pi/3)$, independent of n . According to Lemma 4, we prove the theorem for L_2 if we show that this value is the maximum of

$$\bar{\eta}(\alpha_0) := \lim_{n \rightarrow \infty} \bar{\eta}_n(\alpha_0).$$

This $\bar{\eta}(\alpha_0)$ is the L_1 equivalent of the objective function $\bar{\gamma}(\alpha_0)$.

The singular points of $\bar{\eta}_n(\alpha_0)$ are $k \cdot \frac{(\pi/2)}{2^n}$, with $k = 0, \dots, 2^n$. The lobes in between are characterised by the first n digits in the binary representation of $\beta_0 = \alpha_0/(\pi/2)$, as indicated in Figure 10. As n increases, the plot of $\bar{\eta}_n(\alpha_0)$ soon becomes chaotic.

$\alpha_0 = \pi/3$ clearly is not a maximum of $\bar{\eta}_2$ or $\bar{\eta}_3$. Especially starting angles near 0 or $\pi/2$ seem to show a much slower convergence than that of $\alpha_0 = \pi/3$. This is however just an initial behaviour. For the asymptotic analysis, it suffices to analyse within the interval $(\pi/4, 3\pi/8)$, but even there, $\pi/3$ is not a maximum. Even within its own lobe, there are angles that converge more slowly in the first three steps.

Nevertheless, we can prove the following lemma:

Lemma 5 Call a_n and b_n the end points of the lobe containing $\pi/3$ in the n -th subdivision step, i.e., $\lim_{\alpha \rightarrow a_n} \overline{\eta}_n(\alpha) = -\infty = \lim_{\alpha \rightarrow b_n} \overline{\eta}_n(\alpha)$ and $\overline{\eta}_n(\alpha) \in \mathbb{R}, \forall a_n < \alpha < b_n$. Call $m_n = (a_n + b_n)/2$ the middle point of this lobe and call $\widehat{\eta}_n(\alpha) = \overline{\eta}_n(2m_n - \alpha)$ the mirror of $\overline{\eta}_n$ around this center point. Then:

$$\forall \alpha \in (a_n, m_n) : \overline{\eta}_n(\alpha) > \widehat{\eta}_n(\alpha) \text{ iff } a_n < \pi/3 < m_n.$$

In other words, every lobe that contains $\pi/3$ is asymmetric around its center, and the side that contains $\pi/3$ is always higher than its mirror. Moreover, two mirroring angles α_n and $\widehat{\alpha}_n$ within the same lobe in step n satisfy:

$$\widehat{\alpha}_n = \frac{\pi}{2} - \alpha_n,$$

and according to Lemma 3, the subsequent subdivision steps lead to exactly the same L_1 error decay for both α_n and its mirror.

As a consequence, if for instance

$$\alpha_0 = .101010010010101 \dots \frac{\pi}{2},$$

then one can always find an initial angle with slower convergence, namely its mirror in the lobe 101010:

$$\alpha_0^{[1]} = .101010101101010 \dots \frac{\pi}{2}.$$

As soon as two consecutive digits in the binary notation of β_0 are equal, flip all subsequent digits, and the resulting angle will converge slower. Repeating this procedure on any arbitrary α_0 leads to $\alpha_0 = \pi/3$ being the slowest possible initial configuration.

A similar situation appears in [4, Sections 7.1.1, 7.1.2] in a totally different context. Our problem, however, does not satisfy the assumptions in Lemmata 7.1.6 and 7.1.7 of [4].

Proof of Lemma 5:

First note that the midpoint of the lobe containing $\pi/3$ satisfies:

$$m_n = \pi/3 + \Delta_n,$$

with

$$\Delta_n = (-1)^{n+1} \frac{\pi}{12 \cdot 2^n}.$$

The end points are the midpoints from the previous two steps. If n is even, then $a_n = m_{n-2}$ and $b_n = m_{n-1}$. If n is odd, we have $a_n = m_{n-1}$ and $b_n = m_{n-2}$. The rest of the proof concentrates on the n even case. The odd case is completely similar.

Second, the average log error reduction for angles in the $\pi/3$ -lobe can be simplified to:

$$\overline{\eta}_n(\alpha) = \frac{1}{n} \log \left(\prod_{k=0}^{n-1} \frac{1}{\tan(2^k \alpha)} \cdot \prod_{k=1}^n \frac{1}{\tan(2^k \alpha)} \right)$$

and its derivative equals:

$$\overline{\eta}_n'(\alpha) = -\frac{1}{n} \sum_{k=1}^n \left(\frac{2^k}{\sin(2^k \alpha)} + \frac{2^{k+1}}{\sin(2^{k+1} \alpha)} \right).$$

We show that for all even n :

1. $\overline{\eta}_n'(m_n) > 0$,
2. $\overline{\eta}_n'(\alpha) - \widehat{\eta}_n'(\alpha)$ has no zero,

which is sufficient to prove the lemma. Note that $\widehat{\eta}_n'(\alpha) = -\overline{\eta}_n'(2m_n - \alpha)$.

We proceed by induction. The main difficulty is that the interval on which we consider $\overline{\eta}_n$ changes too, so let us set:

$$x = 2^{n-2}(\alpha - m_n),$$

and

$$\begin{aligned} f_n(x) &:= n \overline{\eta}_n'(2^{-n+2}x + m_n) \\ &= - \sum_{k=1}^n \left(\frac{2^k}{\sin(2^k(2^{-n+2}x + m_n))} + \frac{2^{k+1}}{\sin(2^{k+1}(2^{-n+2}x + m_n))} \right) \\ &= - \sum_{i=0}^{n-1} 2^{n-i} \left(\frac{1}{\sin(2^{2-i}x + 2^{n-i}\frac{\pi}{3} + 2^{n-i}\Delta_n)} + \frac{2}{\sin(2^{3-i}x + 2^{n-i+1}\frac{\pi}{3} + 2^{n-i+1}\Delta_n)} \right) \\ &= - \sum_{i=0}^{n-1} 2^{n-i} \left(\frac{1}{\sin(2^{2-i}x + 2^{n-i}\frac{\pi}{3} + 2^{2-i}\Delta_2)} + \frac{2}{\sin(2^{3-i}x + 2^{n-i+1}\frac{\pi}{3} + 2^{3-i}\Delta_2)} \right) \end{aligned}$$

Note that $\sin(2^{m+2}\pi/3 + u) = \sin(2^m\pi/3 + u)$, so the first $n-1$ terms in the expression for $f_{n+2}(x)$ are equal to $4f_n(x)$ and hence

$$\begin{aligned} f_{n+2}(x) = 4f_n(x) &- 4 \left(\frac{1}{\sin(2^{2-n}x + 4\frac{\pi}{3} + 2^{2-n}\Delta_2)} + \frac{2}{\sin(2^{3-n}x + 8\frac{\pi}{3} + 2^{3-n}\Delta_2)} \right) \\ &- 2 \left(\frac{1}{\sin(2^{1-n}x + 2\frac{\pi}{3} + 2^{1-n}\Delta_2)} + \frac{2}{\sin(2^{2-n}x + 4\frac{\pi}{3} + 2^{2-n}\Delta_2)} \right) \end{aligned}$$

If we define

$$H(u) = -4 \left(\frac{1}{\sin(4u + 4\frac{\pi}{3})} + \frac{2}{\sin(8u + 8\frac{\pi}{3})} \right) - 2 \left(\frac{1}{\sin(2u + 2\frac{\pi}{3})} + \frac{2}{\sin(4u + 4\frac{\pi}{3})} \right)$$

we have

$$f_{n+2}(x) = 4f_n(x) + H\left(\frac{x + \Delta_2}{2^n}\right),$$

where $f_0(x) = 0$.

This allows to fill in the values we need:

1. $\overline{\eta}_n'(m_n) = f_n(0) = 4f_{n-2}(0) + H\left(\frac{\Delta_2}{2^{n-2}}\right)$. So, $f_2(0) = H(\Delta_2) \approx 1.4889$. Since $H(u)$ is monotonously decreasing on $[\Delta_2, 0]$, we can construct a minorant for $f_n(0)$:

$$\hat{f}_n = 4\hat{f}_{n-2} + H(0).$$

It is easy to prove that all elements in \hat{f}_n are positive, and hence so is $f_n(0)$. This proves that $\overline{\eta}_n'(m_n) > 0$.

2. Next, we compare the derivative of the mean log decay with its mirror:

$$\begin{aligned} \overline{\eta}_n'(\alpha) - \widehat{\eta}_n'(\alpha) = f_n(x) + f_n(-x) &= (f_{n-2}(x) + f_{n-2}(-x)) \\ &+ \left(H\left(\frac{x + \Delta_2}{2^{n-2}}\right) + H\left(\frac{-x + \Delta_2}{2^{n-2}}\right) \right) \end{aligned}$$

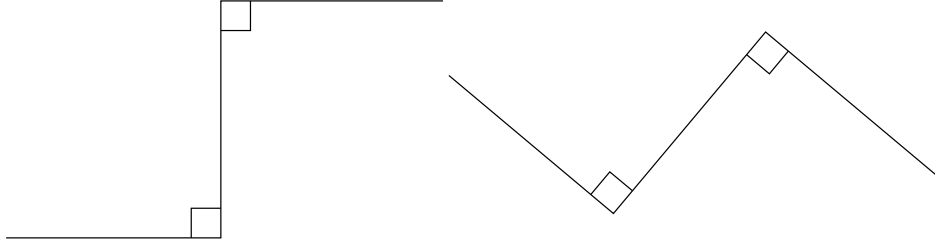


Figure 11: A normal polyline does not make any distinction between piecewise constant functions (left) and continuous piecewise linear functions with rectangles in the singular points (right). As a matter of fact, the notion of function is not that important at all.

We want to prove that this never has a zero. It can be proven that the first two expressions, for $n = 2$ and $n = 4$ are always positive with a minimum in $x = 0$. In order to extend this to arbitrary n , we first note that for even $n > 4$

$$H\left(\frac{x + \Delta_2}{2^{n-2}}\right) + H\left(\frac{-x + \Delta_2}{2^{n-2}}\right)$$

reaches its (negative) minimum in the end points $x = \pm\pi/16$. Second, this minimum is bounded by $2H(0)$. (All this requires plots and/or intensive calculations.) This allows to construct a positive minorant sequence for $f_n(x) + f_n(-x)$, from which it follows that $\overline{\eta}_n'(\alpha) - \widehat{\eta}_n'(\alpha)$ is strictly positive on the entire lobe.

This concludes the proof of Lemma 5 and hence of Theorem 2. □

4 Normal polylines for 1-d piecewise linear and piecewise smooth functions

Unlike the wavelet approach, a normal polyline approximation does not use basis functions. It even does not rely on any concept of function whatsoever. A jump in a piecewise constant function leads to the same (L_1) error decay as a rotated version of this function, which is a continuous, piecewise linear with a rectangle in the singular points, as in Figure 11.

The question arises how well normal polylines do the job for general piecewise linear functions. Figure 12 analyses the performance in a singular point with angle θ (sometimes referred to as a *cuspl*). Suppose that the normal subdivision point reduces S_j and not H_j , as in the Figure. It holds that $\alpha_{j+1} = \pi - 2(\pi/2 - \alpha_j) = 2\alpha_j$, and since α_{j+1} and θ are the angles of a triangle, this imposes the condition that $\alpha_j < (\pi - \theta)/2$. Otherwise, the new point reduces H_j and not S_j . From Figure 12, we see that

$$\tan \alpha_{j+1}(S_{j+1} - \cos \theta H_j) = \tan \alpha_j(S_j - \cos \theta H_j),$$

and

$$\frac{H_j}{\sin \alpha_j} = \frac{S_j}{\sin(\pi - \theta - \alpha_j)}.$$

This leads to:

$$S_{j+1} = \left(\frac{\cos \theta \sin \alpha}{\cos \theta \sin \alpha + \cos \alpha \sin \theta} + \frac{\cos \alpha \sin \theta}{\cos \theta \sin \alpha + \cos \alpha \sin \theta} \frac{\tan \alpha}{\tan 2\alpha} \right) S_j.$$

If $\alpha_j > (\pi - \theta)/2$, we have that $\alpha_{j+1} = 2\alpha_j - (\pi - \theta)$.

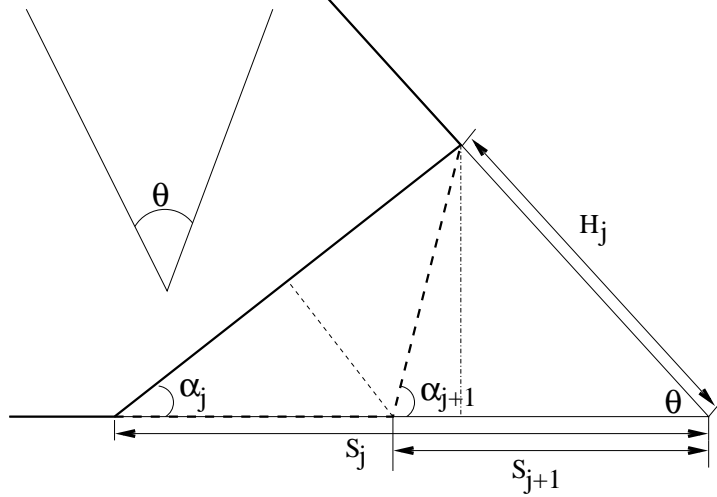


Figure 12: Multiscale normal polyline approximation of piecewise linear function with angle θ . This function (upper left) can be “laid down” to analyse the L_1 error reduction in the j -th step.

Just as in the rectangular case, we can write α_j as a fraction of the maximal angle, $\alpha_j = \beta_j(\pi - \theta)$. If $\beta_0 = 1/3$ or $\beta_0 = 2/3$, all subsequent β_j alternate between these two values. If $\beta_j = 1/3$, i.e., $\alpha_j = (\pi - \theta)/3$, we have $2\alpha_j = \pi - \theta - \alpha_j$, and so, from the expressions above, we find

$$\frac{H_j}{\sin \alpha_j} = \frac{S_j}{\sin(2\alpha_j)} \Rightarrow \frac{H_j}{S_j} = \frac{1}{2 \cos(\alpha_j)} = \frac{1}{2 \cos(\frac{\pi - \theta}{3})}$$

Moreover, $S_{j+1}/S_j = (H_j/S_j)^2$, so the corresponding L_1 -norm reduction in this j -th and all other steps equals:

$$\frac{S_{j+1}}{S_j} = \frac{1}{4 \cos^2(\frac{\pi - \theta}{3})}.$$

For small θ , this error rate can come arbitrarily close to 1. At $\theta = \pi/4$, normal polylines catch a singularity at the same rate as wavelets. But even for smaller θ , the approximation of a singularity requires only one coefficient at each scale. This is due to the locality property, which this approach shares with the classical wavelets. As a consequence, the effort for the approximation of isolated singularities vanished when compared to approximating a superposed smooth function.

5 A few words on normal polyline approximation of smooth functions

The convergence rate of a normal polyline approximation for smooth functions (i.e., functions with a certain number of derivatives) has been extensively investigated [5, Sections 5 and 6]. For further reference, we list a few results which are special cases of that general analysis.

Lemma 6 *If a function f has a bounded derivative on the interval $[a, b]$, then the subdivision points $x_{j,k}$ at scale j and location k found by normal offsets on linear prediction, starting from a (strictly) increasing sequence $\{x_{0,k}\}_k$, satisfy:*

$$\begin{aligned} |x_{j+1,2k+1} - x_{j+1,2k}| &\leq \delta \cdot |x_{j,k+1} - x_{j,k}| \\ |x_{j+1,2k+2} - x_{j+1,2k+1}| &\leq \delta \cdot |x_{j,k+1} - x_{j,k}|, \end{aligned}$$

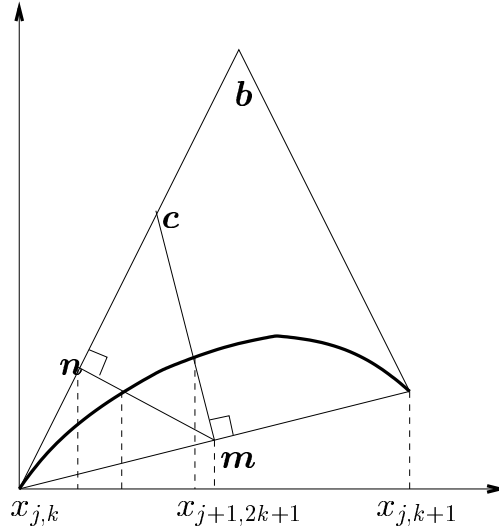


Figure 13: Proof of Lemma 6. Call $\mathbf{m} = (x_m, y_m) = ((x_{j,k} + x_{j,k+1})/2, (f_{j,k} + f_{j,k+1})/2)$ the prediction point and call $M = \sup |f'(x)|$. Then construct two lines with negative and positive slope M , meeting in point \mathbf{b} . By construction, this polyline is a majorant for f . As a consequence, the intersection $\mathbf{c} = (x_c, y_c)$ of this polyline with the normal direction satisfies: $|x_c - x_m| > |x_{j+1,2k+1} - x_m|$. A further upperbound can be found by replacing the true normal direction with the lowest possible slope, i.e., the normal on the polyline in \mathbf{b} . The intersection $\mathbf{n} = (x_n, y_n)$ satisfies $|x_n - x_m| > |x_{j+1,2k+1} - x_m|$. For finite M , x_n cannot be arbitrarily close to either $x_{j,k}$ or $x_{j,k+1}$, so $|x_{j+1,2k+1} - x_m| \leq \delta' |x_{j,k} - x_m|$ with $\delta' < 1$. Now it follows that $|x_{j+1,2k+1} - x_{j,k}| \leq \delta |x_{j,k+1} - x_{j,k}|$ with $\delta = (\delta' + 1)/2$.

for some $0 < \delta < 1$, only dependent on f , not on j .

This means that

$$\limsup_{j \rightarrow \infty} |x_{j,k+1} - x_{j,k}| \rightarrow 0,$$

i.e., the normal offsets, though not so regular as classical subdivision, leaves no ‘gaps’, or unrefined subintervals.

The proof relies on a geometrical construction in Figure 13.

Lemma 7 Suppose a function f has a bounded second derivative on the interval $[0, h]$, and fix $f(0) = 0$, and call $x(h)$ the x -coordinate of the normal subdivision point, then

$$\lim_{h \rightarrow 0} \frac{x(h)}{h} = \frac{1}{2} = \lim_{h \rightarrow 0} \frac{h - x(h)}{h}$$

This means that normal refinement generates asymptotically regular subdivision grids.

Proof: The linear interpolation in $(0, 0)$ and $(h, f(h))$ has an approximation error $|E(x)| \leq Mh^2/2$, with $M = \sup |f''(x)|$. This holds for all $x \in [0, h]$, in particular for x the subdivision point, as illustrated in Figure 14. By Pythagoras’ theorem, we have for $\epsilon := x - h/2$ that $|\epsilon| < |E| \leq Mh^2/2$. So,

$$\lim_{h \rightarrow 0} \frac{x}{h} = \lim_{h \rightarrow 0} \frac{h/2 + \epsilon}{h} = 1/2.$$

□

Corollary 1 If f has a bounded second derivative on the interval $[a, b]$, then the approximation error of a normal polyline f_j after j refinement steps satisfies:

$$|f_j(x) - f(x)| \leq C2^{-2j},$$

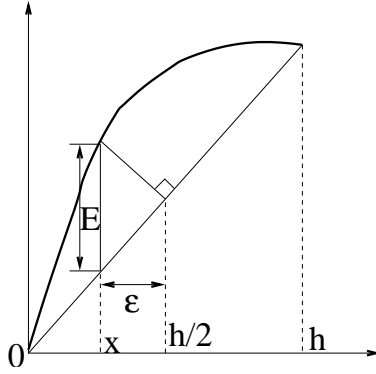


Figure 14: At fine scales, normal subdivision creates nearly regular grids.

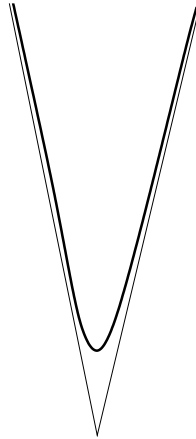


Figure 15: Although asymptotically normal polylines on smooth data perform as well as regular subdivision on the x -coordinate, initial convergence of this smooth function may be as slow as the worst case analysis of the piecewise linear function in Figure 12.

for some C dependent on f but independent of j .

This means that for certain normal subdivision has the same approximation rate as classical, regular subdivision. This follows from Lemmata 6 and 7: first, we are sure that the length of all subintervals converge to zero, and because they do so, the second lemma guarantees that all subintervals have length of order 2^{-j} . The result then follows from the fact that linear interpolation converges quadratically if f has a bounded second derivative.

Remark Although the asymptotic convergence of normal polylines shows the same rate as the convergence for regular subdivision on the x -coordinate, the initial convergence could be substantially slower. This happens if a smooth function is well approximated by a sharp cusp as in Figure 15. The initial convergence behaves as described in the worst case analysis of Section 4. From practical point of view, it is crucial to find initial points on the ‘sharpest’ cusps in the function. In a sense, these points with highest second derivative bear the essential information of the smooth curve. In the following sections, we consider smooth horizons in 2-d, i.e., smooth line singularities. The analysis in this section has illustrated that placing initial points near highly curved parts of such a line singularity is an interesting choice.

6 Analysis II: Normal Meshes for 2-d Piecewise Constant Functions

6.1 Horizon class images

This section analyses the adaptivity of the normal offset scheme for so-called *horizon class* images [8], that is, 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.

6.2 2-d topological problems near singularities

As discussed above, adjacency handling in 2-d is non-trivial. Careful procedure design is necessary to deal with topological exceptions which otherwise may slow down the approximation convergence. This section discusses a simple algorithm which may not be optimal with respect to smoothness. It does preserve, however, the singularity approximation potential of a normal mesh scheme.

Adjacency in 2-d is handled by triangulation. The question arises how to proceed at the lines of singularity. At first sight, it might look natural to have vertical triangles between three vertices on the horizon. These triangles constitute a piecewise linear approximation of a vertical, curved surface. Subsequent refinements may cause flipping and mutually crossing triangles when the edges of these vertical triangles are being subdivided: the new piercing points all lie on the horizon (the data singularity), but not necessarily in a consistent way. The reason is that the projection of vertical triangles onto the domain space has area to zero. Hence, this function domain can no longer serve as a parameter or reference space to control the refinement process.

In order to avoid flipping triangles, the parameter domain (in our case domain of the function, i.e., a subset of \mathbb{R}^2) could put limits on how far the algorithm can look in the normal direction. The projection of the piercing point onto the parameter domain should be such that the refined triangular grid shows no overlaps (crossing edges) in this reference domain. If such a piercing point cannot be found in the normal direction, the new point should be taken as far as possible in the normal direction. This approach at least partly saves the benefits from a normal offset pointing towards the location of the horizon. A correction coefficient in vertical direction is then needed to find the corresponding function value. Figure 16(b) has the 1-d version of this case: starting from a midpoint m we are not allowed to find piercing points further than n (which in 1-d obviously does not make any sense), so we continue to look in the vertical direction, leading to point v . In this case only the vertical offset needs to be stored, together with a bit indicating that we first have to go up to the acceptance limit. Most, though not all, conflicts are avoided if we divide each triangle *a priori* in three regions for every edge to be subdivided, as in Figure 16(a). An additional check deals the few remaining exceptions, i.e., configurations where edges $e_1 f_2$ and $e_2 f_2$ are subdivided with two new close to, respectively e_1 and e_2 , while the third new vertex would be at the location of n in Figure 16(a).

6.3 Polyline approximation of the horizon

Working with vertical triangles on singularities poses at least two problems:

1. *Practically* lots of narrow, arbitrarily oriented triangles appear near the data discontinuity.
2. *Theoretically* all the complicated exception handling may and probably will destroy the 1-d fast convergence. Because of their intractability, vertical triangles are excluded, thereby giving up their intrinsic good approximation of horizons.

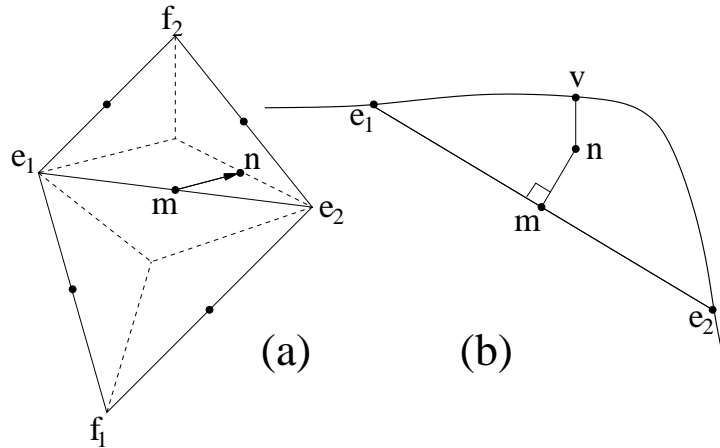


Figure 16: (a) Putting limits to how far one can search in the normal direction. This is controlled by the reference domain. The dotted lines indicate the boundaries in which piercing points are accepted. (b) A 1-d version of the procedure followed when no normal piercing point is found in the acceptance region

Since triangulation is not a goal as such, an alternative is to provide two function values, a lower and upper value, in each vertex. The goal is now a gradual improvement of these values as an approximation of the true discontinuity, just as in the 1-d case. We consider three types of edges in the triangulation: the first class contains edges that connect two horizon vertices. The second class connects a horizon with a non-horizon vertex, whereas the third class does not interfere with the horizon. Subdivision of edges in the first class should lead to a new horizon vertex. To maintain control over this process, we look for a piercing point in a direction normal to the vertical plane containing these two horizon vertices. This means that the slope of the edge connecting them is not taken into account.

Subdivision of edges in the second class (connecting a horizon point with a non-horizon vertex) should *not* lead to a new horizon vertex, but rather to a better approximation away from the horizon. We look for a piercing point which lies *in* the vertical plane containing that edge and the slope of that edge is the *only* degree of freedom to be filled in when computing the normal direction. Either this new vertex does or does not coincide with the existing end point on the horizon. If it does not, no special action has to be taken: we just insert the newly found vertex. If it does, we update one of the two function values in the existing point, just as in the 1-d case. We also insert a new vertex at the midpoint of the edge for two reasons. First, for simplicity, we want to subdivide every triangle into four children. Second, this avoids long and skinny triangles perpendicular to the horizon. We store the vertical offset between the function value in this vertex and its prediction on the subdivided edge as a detail coefficient, together with a label indicating that this is not a normal but vertical offset. We do not store the normal offset, since the label already tells the reconstruction algorithm that it should go as far as possible, up to the previously introduced vertex.

Edges in the third class do not need a special treatment.

6.4 Straight horizons

Before we proceed with algorithmic details, let us take a moment to study the behaviour of this algorithm on straight line horizons. As soon as two points on the horizon have been detected, the only error comes from triangles with one edge on the horizon. By construction, the area of such a triangle is always less than a quarter of its parent: the parent has four children and the refinement points are always closer to the horizon than half of the total edge length. As a consequence, the total area of the error zone is reduced by a factor more than two in every step. Moreover, all edges that link a horizon point with a non-horizon point inherit the 1-d property that the error height H_j after j refinements has the same order of magnitude as the edge

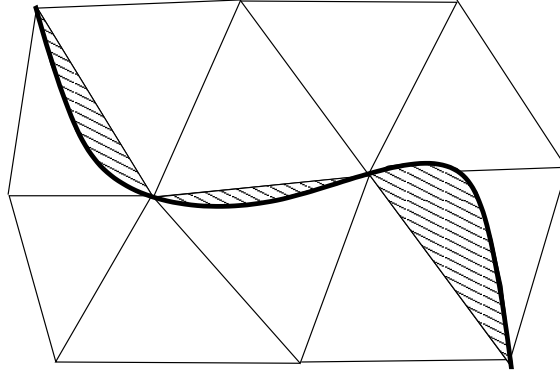


Figure 17: *Normal mesh approximation in 2-d: the shaded area corresponds to the dominant term in the approximation error.*

length S_j :

$$H_j = \mathcal{O}(S_j).$$

As a consequence, the total squared L_2 -error satisfies:

$$\|f - f_j\|^2 = \mathcal{O}(2^{-3j}),$$

to be compared to

$$\|f - f_j\|^2 = \mathcal{O}(2^{-j})$$

for a wavelet approximation. The gain lies in the simultaneous reduction of error width and height.

6.5 Smooth horizons

If the horizon is a smooth curve (that is, at least differentiable with respect to local coordinates), then the algorithm aims at a polyline approximation of this curve. We then obtain the situation in Figure 17.

All triangles shown in this picture are in contact with the singularity and therefore they are not entirely flat in 3-d. Nevertheless, thanks to the fast 1-d error decay, these triangles rapidly approach the horizontal surface which they are supposed to approximate. In the shaded area however, between the horizon and its polyline approximation, the approximation of the 2-d function f is essentially incorrect.

This term turns out to be dominant, so it is crucial to keep it under control. In order to do so, the algorithm must keep track of *where* the horizon is at all times and without (too much) overhead.

At the analysis (decomposition), singularities can be detected when finding piercing points: if the piercing surface is vertical (in practical applications: when it is “very steep”), we have a horizon point. At the reconstruction, the only way we can detect horizon points from the data themselves is by finding the vertices with two different function values. A newly found horizon point however has only one function value in its first step of existence. If a horizon vertex follows from the refinement of an edge in the approximating polyline, this can be monitored in the reconstruction phase as well.

Since triangle edges near the horizon are subdivided in a special way, it is important to:

1. ensure that refinement of the horizon approximation leads to new horizon points as often as possible
2. detect new horizon points immediately if they do not originate directly from the horizon approximation.

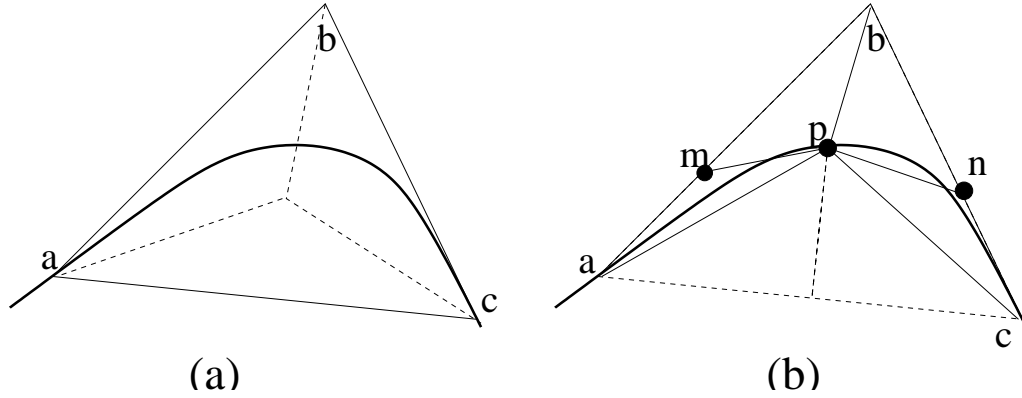


Figure 18: (a) Triangles near the polyline approximation (ac) of the horizon need special treatment in order to keep track of the location of the horizon. Refinement of the polyline approximation should lead to a closer approximation. Without special action, the true horizon may fall outside control of the approximating edge. (b) Therefore we give absolute priority to refinement of the approximating edge.

A refinement of an edge connecting two horizon points should lead to new horizon point, if possible. The procedure as it is now, does not always let the new point go far enough. Figure 18(a) illustrates what may happen near sharp curvatures. Since the search field within a triangle is divided into three regions for each edge, the real edge may fall outside the control of its polyline approximation (ac). Therefore, we modify the procedure, such that in triangles containing the polyline approximation, the polyline edge has absolute priority. This is shown in Figure 18(b). Also, retriangulation after refinement is different: normally, the new vertices are connected with each other and with the end points of the edge that they subdivide. In Figure 18(b)), however, the edge mn would have intersections with edges ap and cp . Therefore, we modify the retriangulation algorithm for triangles in which a new horizon vertex is found on a normal search direction: if necessary, the new horizon vertex is connected to its opposite, old vertex (leading to edge bp in Figure 18(b)).

Figure 18(b) also illustrates the second issue: connecting a new horizon vertex with its neighbours may result in two edges crossing the horizon. This poses no problem for edges that are part of the polyline approximation, but other edges crossing the horizon, like pn and pm in the figure, cause new horizon points which are not a result of the polyline refinement procedure. It is acceptable for the algorithm to store the vertex numbers of the first points of a newly found horizon. From then on, the algorithm should be able to reconstruct the horizon from the coefficients only.

To this end we use the bit that normally indicates whether a coefficient is normal or vertical. If an edge connects two horizon points, the normal search direction was taken horizontal, as discussed before. If no intersection point is found, this means that the horizon has at least one intersection with at least one of the two remaining edges. So, if the search label tells the reconstruction algorithm to look in vertical direction, we are in one out of three situations, depicted in Figures 19, 20, and 21. A new horizon point may also show up while the present polyline approximation can still be subdivided in a normal direction. See Figure 22. The next step however immediately leads to the situation in Figure 20.

The two labels of the remaining edges in this triangle indicate in which situation we are: is the third vertex (b) at the opposite side of the horizon, and if so, do we have just one new horizon vertex (as in Figure 19), or has the horizon intersections with both edges. If there is just one intersection (first case), this must be on the longer edge. Indeed, otherwise the polyline approximation of the horizon would have been refined in this step (vertex p in Figure 19 would have been on the horizon.)

The second possibility is Figure 20, where the two other edges should be refined as polyline approximations of the horizon. Note that in Figure 20, a new exception for the next step is generated promptly in vertex q .

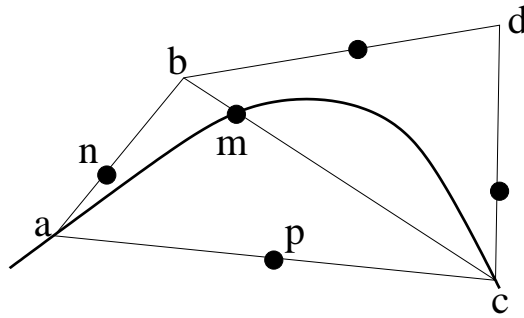


Figure 19: *First possible configuration where horizon vertices are found outside the refinement process of the polyline approximation of the horizon. At the reconstruction, this configuration is detected by the fact that the horizon approximating edge (ac) is not subdivided in a normal direction.*

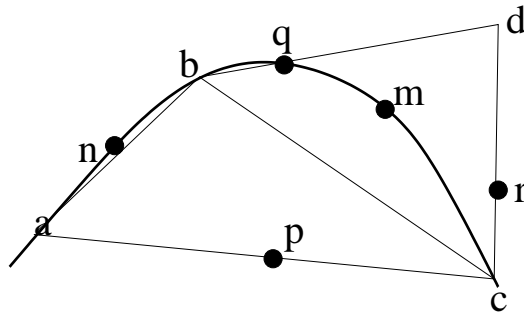


Figure 20: *Second configuration with horizon vertices not following from refinement of the polyline approximation (ac) of the horizon. Vertex b was probably an unforeseen, additional horizon vertex in the previous step, but only now this has to be taken into account.*

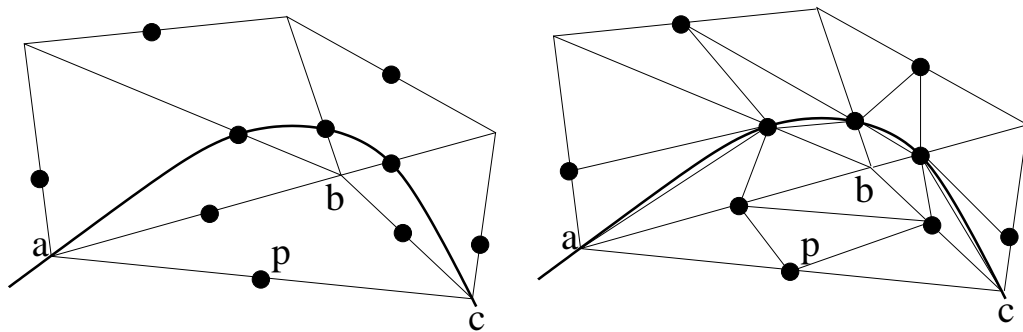


Figure 21: *Third configuration with horizon vertices not following from refinement of the polyline approximation (ac) of the horizon.*

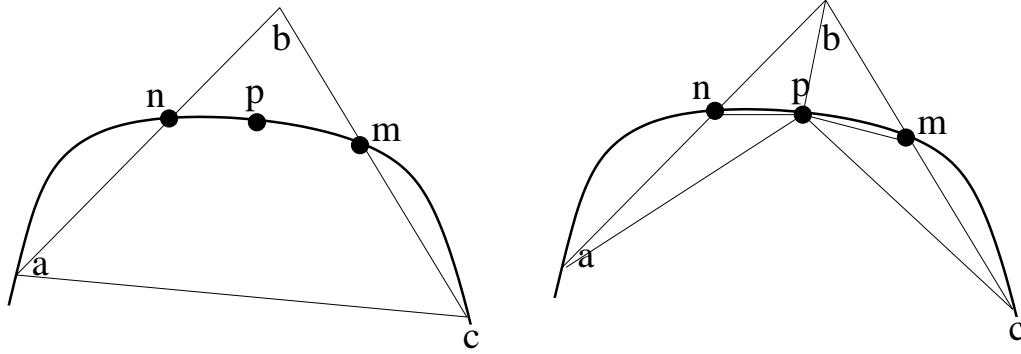


Figure 22: *Fourth configuration with horizon vertices not following from refinement of the polyline approximation of the horizon. Unlike the first three, this one cannot be detected from the refinement action on the horizon approximating edge. Nevertheless, subdivision immediately leads to the situation in Figure 20.*

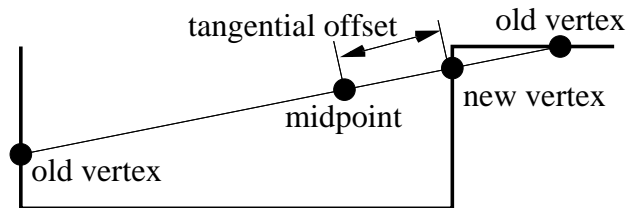


Figure 23: *Tangential refinement. Good for fast location of singularities, not appropriate for further convergence.*

The last case, as depicted in Figure 21 is the most complicated one. The approximating polyline segment belongs to two triangles. If these triangles both lie on the same side of the horizon, this means we are in the situation of Figure 21. A simple examination of the actual function values in neighbours of vertex b reveals which triangle should receive a special treatment: all edges of neighbouring triangles that contain b now have an intersection with the horizon.

In all three cases, to make sure that the algorithm locates the intersection immediately, we search along the edge itself instead of going normal. The detail coefficient is now a *tangential* offset. Figure 23 shows what happens in 1-d. Note that this tangential search is fast in *locating* the singularity. Edges that are marked as not having an intersection with the horizon are subdivided in a vertical direction. The question arises whether this combination of tangential offsets for edges and vertical offsets elsewhere could serve as the algorithm as such. This however would be poor for further *convergence* near the singularity. The use of tangential offsets is limited to these exceptional cases.

6.6 Convergence behaviour

We now state the main result of this paper:

Theorem 3 *Given a horizon class function f defined on $[0, 1] \times [0, 1]$, or on any compact subset of \mathbb{R}^2 . then under mild assumptions the normal mesh approximation f_n with n non-zero offsets converges in L_2 at a rate of*

$$\|f - f_n\| = \mathcal{O}(n^{-1}),$$

Before starting the actual proof, we explain what is meant by *mild assumptions*.

Assumption 1 For ease of notations, we assume that the horizon can be described as a smooth function $y = H(x)$. Obviously, since normal offsets have no natural restriction towards functions, any curve with the same smoothness will work, even if it cannot be written in an explicit function form.

Assumption 2 By a smooth horizon, we mean a twice differentiable function $y = H(x)$ with a bounded second derivative. In practice, this means we exclude curves with infinitely sharp cusps.

Assumption 3 We assume that the normal refinement procedure with the exception handling from Section 6.5 creates an initial mesh in which the horizon is approximated by a polyline. Call h the polyline resolution, i.e., the length of the longest segment of the polyline. We assume that h is sufficiently small. Also, we assume that the altitude of triangles on the polyline is sufficiently large and that none of these triangles has an edge tangential to the horizon. What we mean by ‘sufficiently large’ and ‘sufficiently small’ will be clarified in the proof itself.

Alternatively, we could assume that the initial mesh satisfies this condition, regardless of how it was created.

Proof of Theorem 3:

The proof consists of two parts. The first part is a construction, the second part is the analysis of this construction. We assume that after j refinement steps, using the procedure described in Section 6.5, the triangulation is in a shape satisfying the assumptions. For the proof, we concentrate on what happens with further refinement of one segment of the polyline. We introduce local coordinates, such that the segment coincides with the $y = 0$, and the two vertices are at $(0, 0)$ and $(h_j, 0)$. $h_j \leq h$ is now the resolution of this polyline segment. We denote by $M = \sup_{[0, h_j]} |H''(x)|$.

It follows immediately that $\sup_{[0, h_j]} |H'(x)| \leq Mh_j/2$ and $\sup_{[0, h_j]} |H(x)| \leq Mh_j^2/8$.

In the first instance, we assume that the segment of the true horizon between $(0, 0)$ and $(h_j, 0)$ falls entirely within one triangle (i.e., we assume that the horizon has no intersections with its polyline approximation), and we concentrate on that triangle (see for instance Figure 24). We call α_j and β_j the angles of the triangle in the two horizon vertices, i.e., angle in a and b respectively in Figure 24). Furthermore, $\alpha_{j+1,1}$ and $\beta_{j+1,1}$ are the corresponding angles in the left child triangle (Δanp in Figure 24) and $\alpha_{j+1,2}$ and $\beta_{j+1,2}$ are the corresponding angles in the right child triangle (Δbnq in Figure 24). Assuming that the horizon has no inflection points in this interval, we immediately see that $\alpha_{j+1,1} \leq \alpha_j$ and $\beta_{j+1,2} \leq \beta_j$. Also, $\beta_{j+1,1}$ is maximised if subdivision point p coincides with the midpoint of edge ac (the subdivision point cannot be closer to c .) This maximum is bounded by the angle between the line of maximum derivative, i.e., a slope of $Mh/2$ and edge bc . We assume that the triangle has an altitude which ensures that this slope has an intersection with the overlying edge. This assumption will be further refined in the next step of the proof. A similar argument holds for $\alpha_{j+1,2}$. As a consequence, all horizon angles of subsequent triangles are bounded. We call this upper bound α_{\max} .

The exceptions described in 6.5 all originated from edges having unexpected intersections with the horizon. In order to facilitate the analysis, we replace the exception handling procedures with one single rule, which avoids edges from the subdivision process having intersections with the horizon. This single rule needs some space to work, which will explain why we need a ‘sufficiently small’ polyline resolution and a ‘sufficiently large’ triangle altitude. The exception handling rule we are using for this proof is a *forbidden zone*. The border of this forbidden area is parallel to the polyline segment, and the y -value F_j should accumulate to catch the following effects:

1. The maximum horizon value: $Mh_j^2/8$.
2. The maximum slope over a distance of $Mh_j/2$ (from $h_j/2$ to one of the end points)
3. If the triangle has an obtuse angle in one of its horizon vertices, the maximum angle α_{\max} comes in.

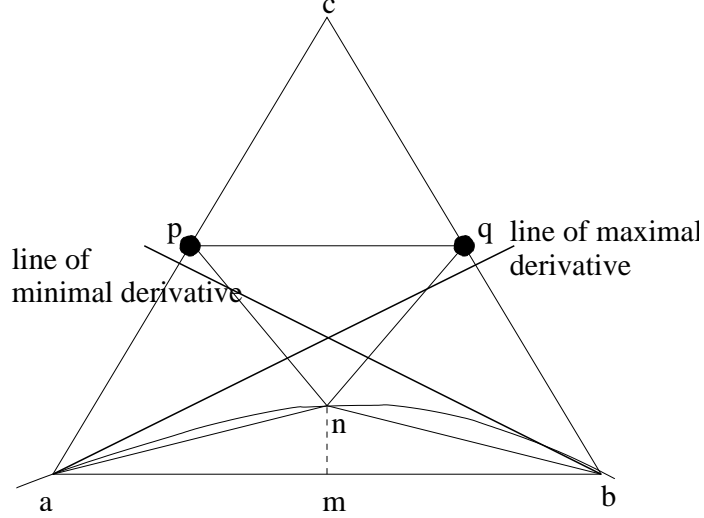


Figure 24: *Triangles containing the horizon cannot have arbitrarily obtuse angles in horizon vertices.*

Call $C_t = 0$ if $\alpha_{\max} \leq \pi/2$ and $C_t = \tan(\alpha_{\max} - \pi/2)$ otherwise, then we take

$$F_j = \frac{3Mh_j^2}{8} \cdot \frac{1}{1 - C_t M h_j / 2}.$$

The second factor corrects for possible obtuse angles. As before, edges connecting a horizon vertex with a non-horizon vertex are subdivided in a 1-d fashion. If the new point has x and y coordinates on the existing vertex, we proceed as before: the approximation value in that vertex is updated, and we also insert a new vertex at the middle point of that edge (vertical offset). If the normal offset however points towards a vertex on the edge in the forbidden zone, we stop at the forbidden border and from there proceed vertically. The edges connecting the new vertex with its neighbours cannot possibly have an intersection with the horizon.

Before proceeding and using this exception handling rule, we must be sure that the construction of a forbidden zone is possible in each successive step. First of all, the altitude of the initial triangle must be at least twice the width of the forbidden strip. Second, this must also be the case in all subsequent steps. The worst case scenario in step j corresponds to a new vertex right on the border of the forbidden strip, see Figure 25. It is clear that the altitude $A_{j+1,1}$ of the child triangle satisfies $A_{j+1,1} \geq F_j - Mh_j^2/8$. So, the condition is $F_{j+1} \leq A_{j+1,1}/2$. Because $\lim_{h_j \rightarrow 0} h_{j+1}/h_j = 1/2$, we can, for sufficiently small h_j choose a constant C arbitrarily close to 1, such that $h_{j+1} \leq Ch_j/2$ and we know for certain that $C \leq 2$, so we can write:

$$\begin{aligned} F_{j+1} &= \frac{3Mh_{j+1}^2}{8} \frac{1}{1 - C_t M h_{j+1} / 2} \\ &\leq \frac{3Mh_{j+1}^2}{8} \frac{1}{1 - C_t M h_j / 2} \\ &\leq \frac{C^2 3Mh_j^2}{4 \cdot 8} \frac{1}{1 - C_t M h_j / 2} \\ &= \frac{3C^2}{4} \frac{Mh_j^2}{8} \frac{1}{1 - C_t M h_j / 2} \\ &= \frac{3C^2}{4} \left(\frac{3Mh_j^2}{8} - \frac{Mh_j^2}{8} \right) \frac{1}{2} \frac{1}{1 - C_t M h_j / 2} \\ &\leq \frac{3C^2}{4} \left(\frac{3Mh_j^2}{8} \frac{1}{1 - C_t M h_j / 2} - \frac{Mh_j^2}{8} \right) \frac{1}{2} \end{aligned}$$

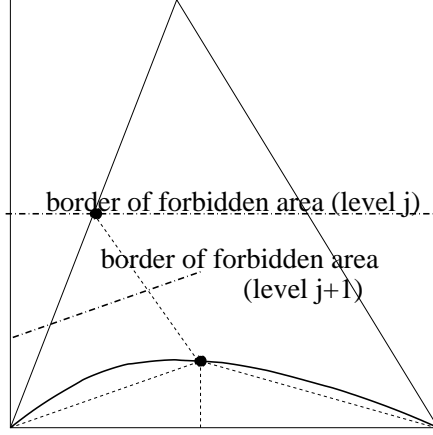


Figure 25: At sufficiently fine scale, it is possible to replace all exception handling with one simple rule, i.e., the construction of a forbidden area, in which no new vertex is allowed. This area does not contain the present polyline segment, so refinements on existing vertices is still allowed. This forbidden area prevents all exceptions in further steps, and makes the asymptotic analysis easier.

$$\begin{aligned}
&= \frac{3C^2}{4} \left(F_j - \frac{Mh_j^2}{8} \right) \frac{1}{2} \\
&= \frac{3C^2}{4} \frac{A_{j+1,1}}{2}
\end{aligned}$$

The construction is therefore possible as soon as $3C^2/4 \leq 1$, which is the case for h_j sufficiently small.

Figure 17 illustrates that two effects determine the convergence rate of the normal offset approximation:

1. the polyline approximation of the edge, and
2. the approximation near that polyline.

We call f_j the approximation of f after j refinement steps (i.e., at level j) and $f_j^{[1]}$ the horizon class function with horizon the polyline approximation after j refinement steps of the horizon in f . We then have for the L_2 -error norm:

$$\|f - f_j\|^2 \leq \|f - f_j^{[1]}\|^2 + \|f_j^{[1]} - f_j\|^2.$$

The actual proof (the second part) now consists of three analyses:

1. The asymptotic behaviour of the polyline approximation of the horizon.
 Since the forbidden area prevents any exception, the polyline approximation of the horizon reduces to what happens in 1-d with a normal polyline approximation of a smooth curve. If we call H_j the polyline approximation of horizon H , then according to Corollary 1 we have $|H_j(x) - H(x)| \leq C2^{-2j}$. As a consequence,

$$\|f - f_j^{[1]}\|^2 = \mathcal{O}(2^{-2j}).$$
2. Triangles lying on the convex side of the horizon, with two vertices on the horizon. The analysis for these triangles starts from what we know about straight horizons. In Figure 26, we know by construction that the area $|\Delta pma|$ of triangle Δpma satisfies $|\Delta pma| \leq |\Delta abc|/4$. The correction for the non-straight horizon leads to $|\Delta pma| \leq |\Delta abc|(1 + M2^{-2j})/4$. If we denote by A_j the total area of triangles with two vertices on the horizon, we have that $A_{j+1} \leq A_j(1 + M2^{-2j})/2$. Since the correction vanishes as j increases, we can write

$$A_j = \mathcal{O}((2 - \varepsilon)^{-j}),$$

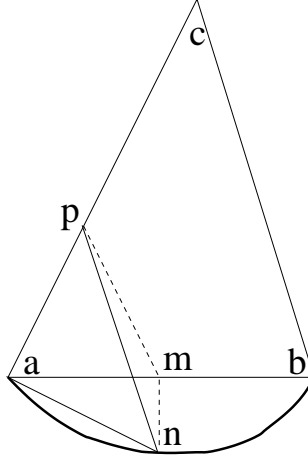


Figure 26: *One refinement at the convex side of the horizon.*

for arbitrary positive ϵ .

3. Triangles lying on the concave side of the horizon, or triangles near an inflection point, have a forbidden area. This zone cannot change the fact that the total area A_j of triangles with two vertices on the horizon satisfies $A_{j+1} \leq A_j/2$.

Just as in the straight horizon case, the error height H_j in a vertex on the horizon after j refinements has the same order of magnitude as the length S_j of the edge connecting this horizon vertex with a non-horizon vertex. This allows to conclude that

$$\|f_j - f_j^{[1]}\|^2 = \mathcal{O}(2 - \epsilon)^{-3j}$$

Putting these elements together, we obtain a convergence rate of:

$$\|f - f_j\|^2 = \mathcal{O}(2^{-2j}).$$

The number n of non-zero coefficients needed for the approximation f_j at level j , satisfies:

$$n = \mathcal{O}(2^j),$$

from which we conclude that the L_2 -error of a nonlinear n -term approximation f_n decays as $\|f - f_n\| = \mathcal{O}(n^{-1})$. □

For reconstruction, the procedure requires storage of one coefficient for each new vertex plus a label telling whether this offset is normal or vertical. We also have to label horizon vertices that are not found by subdivision of an edge connecting two horizon vertices. The number of these “initial” horizon points is of course neglectable.

7 Examples

We now apply the normal offset decomposition to a (digital) image. This image is discrete set of pixels. In order to find normal piercing points, we need to interpolate these pixel matrix. A trivial triangular mesh allows for a piecewise planar interpolation in each point. As a consequence, there is no real discontinuity, only steep transitions. Many edges in images are blurred over several pixels anyway. The special actions to deal with real discontinuities are therefore unnecessary in this practical example.

Figure 27 compares a six level wavelet reconstruction with a six level normal offset reconstruction. For the wavelets, we used the biorthogonal wavelets of Cohen, Daubechies, and Feauveau with two vanishing



Figure 27: Normal offsets (b), compared to classical wavelets (a). CDF 2,2 wavelets, i.e., linear prediction followed by two-taps update (i.e., update with two coefficients). Normal offsets are directed towards the edges. Triangulation allows for sharper edge reconstruction.

moments. The lifted implementation of this wavelet transform is the simplest scheme with a linear prediction operator. It is therefore close to the prediction operator in the normal offset approach. The adaptive triangulation leads to sharper edge reconstruction. For small structures, such as the ideas in the photograph, or for texture, wavelets perform better in filling up the details. This suggests that a combination of both approaches might be interesting in practical applications.

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

Normal offsets are the key to *adaptive* triangulation of 2-d data sets. These data may contain line singularities, posing substantial problems to any tensor product based decomposition. In a normal offset decomposition the multiscale detail coefficients carry information on the location of the line singularities. For horizon class images, this leads to a $\mathcal{O}(n^{-1})$ approximation. The procedure is highly nonlinear. Topological exceptions need to be dealt with carefully.

Current research concentrates on the applicability of the normal offset concept on real images:

1. In practice, a good initial mesh seems to have crucial impact on the performance. The same idea of normal search can be used to select a limited number of crucial, coarsest scale samples (pixels).
2. The nonlinear character of the decomposition itself makes it harder to analyse the effect of removing or modifying a given coefficient. An analysis in 1-d is possible, e.g. in L_1 . In 2-d, the topological exceptions complicate the whole thing: changing a single coefficient may influence the topology on the following, finer grids.
3. 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.
4. This observation also explains why nonlinear approximation, for compression, is a non-trivial task. Thresholding or tree structured coefficient selection has to deal with the topological aspects.
5. In practice, images are obtained as samples on a square grid, so using normal meshes is equivalent to a *remeshing* operation. A second inverse remesh would be necessary to display a normal mesh approximation using a conventional display or printer. This makes things more complicated in practice.

References

- [1] E. J. Candes and D. L. Donoho. Curvelets - a surprisingly effective nonadaptive representation for objects with edges. Technical report, Department of Statistics, Stanford University, 2000.
- [2] R. L. Claypoole and R. G. Baraniuk. A multiresolution wedgelet transform for image processing. In M. A. Unser, A. Aldroubi, and Laine A. F., editors, *Wavelet Applications in Signal and Image Processing VIII*, volume 4119 of *SPIE Proceedings*, pages 253–262, 2000.
- [3] A. Cohen and B. Matéi. Compact representations of images by edge adapted multiscale transforms. In *Proc. IEEE Int. Conf. on Image Proc. — ICIP '01*, Thessaloniki, Greece, 2001.
- [4] I. Daubechies. *Ten Lectures on Wavelets*. CBMS-NSF Regional Conf. Series in Appl. Math., Vol. 61. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.
- [5] I. Daubechies, O. Runborg, and W. Sweldens. Normal multiresolution approximation of curves. Technical report, Dept. of Mathematics, Princeton University; Lucent Technologies, 2002. In Preparation.
- [6] R. A. DeVore, B. B. Jawerth, and V. Popov. Compression of wavelet decompositions. *Amer. J. Math.*, 114:737–785, 1992.
- [7] R. A. DeVore, B. Jawerth, and B. J. Lucier. Image compression through wavelet transform coding. *IEEE Transactions on Information Theory*, 38(2):719–746, 1992.
- [8] D. L. Donoho. Wedgelets: Nearly minimax estimation of edges. *Annals of Statistics*, 27(3):859–897, 1999.
- [9] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, 1994.
- [10] D. L. Donoho, M. Vetterli, R. A. DeVore, and I. Daubechies. Data compression and harmonic analysis. *IEEE Transactions on Information Theory*, 44(6):2435–2476, October 1998.
- [11] N. Dyn, D. Levin, and S. Rippa. Data dependent triangulations for piecewise linear interpolation. *IMA Journal of Numerical Analysis*, 10:137–154, 1990.
- [12] N. Dyn and S. Rippa. Data-dependent triangulations for scattered data interpolation and finite-element approximation. *Appl. Num. Math.*, 12:89–105, 1993.

- [13] I. Guskov, K. Vidimee, W. Sweldens, and P. Schröder. Normal meshes. In *SIGGRAPH 2000 Conference Proceedings*, 2000.
- [14] S. Mallat. Wavelets, approximation, and compression. *IEEE Signal Processing Magazine*, 18(5):59–73, September 2001.
- [15] S. G. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 525 B Street, Suite 1900, San Diego, CA, 92101-4495, USA, 1998.
- [16] I. Niven. *Irrational Numbers*, volume 11 of *Carus Mathematics Monographs*. Mathematical Association of America, 1956.
- [17] S. Rippa. Long and thin triangles can be good for linear interpolation. *SIAM J. Numer. Anal.*, 29(1):257–270, 1992.
- [18] W. Sweldens and P. Schröder. Building your own wavelets at home. In *Wavelets in Computer Graphics*, ACM SIGGRAPH Course Notes, pages 15–87. ACM, 1996.
- [19] R. M. Willett and R. D. Nowak. Platelets: A multiscale approach for recovering edges and surfaces in photon-limited medical imaging. *Submitted*, 2001.