Fast Multiresolution Surface Meshing*

M. H. Gross, R. Gatti, O. Staadt

Computer Science Department
ETH–Zürich, CH–8092 Zürich, Switzerland

Abstract

We present a new method for adaptive surface meshing and triangulation which controls the local level–of–detail of the surface approximation by local spectral estimates. These estimates are determined by a wavelet representation of the surface data. The basic idea is to decompose the initial data set by means of an orthogonal or semi–orthogonal tensor product wavelet transform (WT) and to analyze the resulting coefficients. In surface regions, where the partial energy of the resulting coefficients is low, the polygonal approximation of the surface can be performed with larger triangles without loosing too much fine grain details. However, since the localization of the WT is bound by the Heisenberg principle the meshing method has to be controlled by the detail signals rather than directly by the coefficients. The dyadic scaling of the WT stimulated us to build an hierarchical meshing algorithm which transforms the initially regular data grid into a quadtree representation by rejection of unimportant mesh vertices. The optimum triangulation of the resulting quadtree cells is carried out by selection from a look–up table. The tree grows recursively as controlled by detail signals which are computed from a modified inverse WT.

In order to control the local level–of–detail, we introduce a new class of wavelet space filters acting as "magnifying glasses" on the data.

1. Introduction

Polygonal surface approximations are an essential preprocessing step in scientific visualization, since most modern graphics hardware supports the display of shaded and textured triangles. Nevertheless, in order to treat complex data sets efficiently, methods have to be found to reduce the number of triangles representing the data. This problem is not only striking in the field of digital terrain modeling and flight simulation, but also in many other applications, such as finite element, radiosity [1] or parametric surface meshing [6]. Hence, adaptive triangle reduction techniques were established in the past. Most of them try to find mathematical criteria for the importance of a particular mesh vertex, remove it if applicable and perform a local retriangulation of the mesh. [18] for instance analyzes single vertices in the mesh and defines a planarity criterion to decide on the removal of the vertex. In order to avoid cracks in the surface, a local Delaunay triangulation has to be performed. Quadtree–based methods [17] were proposed mostly for radiosity meshing, where the mesh is controlled by the illumination gradient. Other implementations are used for representing rectangular B–spline patches [6].

Although most of the existing methods work well within the above limitations and can be found in a broad range of applications, the basic issues arising from these approaches are as follows:

I. The criteria employed to thin the triangle mesh are usually based on simple local geometric surface features, such as planarity or Gaussian curvature. It is difficult to quantify global error bounds of the overall approximation.

II. The reduction of the triangle mesh is computationally expensive and once local retriangulations are performed, extensive work on data structures and list management is required.

III. There is no elegant way to focus the level–of–detail locally onto interesting data features – a property of increasing importance in complex data sets.

On the other hand, the wavelet transform, as presented in [5], [13] or [3] has been discovered for computer graphics: [11] and [15] proposed volume rendering techniques, whereas [12] published a volume morphing method. Even approximate solutions of the radiosity equation can be achieved using WTs [8], as well as visualization of multidimensional features, such as in [9]. All of these approaches employ the WT to expand the data and to control the parameters of the approximation within the mathematical bounds of the L^2–energy norm.

The goal of the following paper is to point out an alternative approach to the adaptive triangulation problem: the usage of the wavelet transform as an overall mathematical framework which controls the data approximation.

The concept of our method is illustrated in Fig. 1. The initially regular surface data grid has to be transformed into a quadtree structure and each quadtree cell has to be triangulated using a look–up table. In order to decide, whether a particular mesh vertex can be removed, we first apply a WT onto the data and then iteratively reconstruct the detail signals. The amplitude of the detail signal is taken as a measure for the local frequency characteristics and decides on the removal of points. The dyadic scale of the standard WT reconstructs the detail signals from the different frequency channels in a single step mode. After the first step each second data vertex of the grid is analyzed. Then, as the iteration proceeds the next detail signal is reconstructed and each fourth vertex is analyzed and so forth. This scheme enforces a loop consisting of a single–step inverse WT to recover a particular detail signal and an analysis step to label unimportant coefficients. Applying wavelet space filtering allows an elegant control of the local level–of–detail of the triangulation and acts as a local "magnifier".

Although the scope of our paper is to present a method for 2D surface meshing, it can also be extended to 3D to handle isosurfaces or volumes with tedrahedrizations [2]. Moreover, some of the different ideas encompassed by this method, such as the detail signal criterion and the wavelet space filters can also be used to govern existing meshing methods.

The organization of the paper is as follows: First of all, we describe the mathematical framework of the 2D wavelet transform for surface approximation and particular emphasis is given to the required extensions, such as modifications of the QM–Filter pyramids to figure out the inverse WT. Furthermore, mathematical formulations of filters in wavelet space are explained and their importance for level–of–detail control is stressed. The next section sheds light on the quadtree–based mesh representation we propose and shows how to derive local optimal cell triangulations from a look–up table. The algorithmic complexity of the method as well as an error analysis is elaborated in chapter 4. Finally, some examples from a digital terrain model of the Swiss Alps illustrate the superiority of the proposed method.

2. Surface Approximation using Wavelets

2.1. The 2D Wavelet Transform

The 2D version of the wavelet–transform (WT) expands any finite energy function $f(x,y) \in L^2(\mathbb{R}^2)$ using a set of similar basis functions $\psi_{a,b}(x,y)$. Its generic continuous form description is provided as the following inner product:

$$WT_{f,\psi}(a_x, a_y, b_x, b_y) = \langle f, \psi_{a,b} \rangle = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \psi_{a,b}(x,y)\, f^*(x,y)\, dx\, dy \qquad (1)$$

with $a_x, a_y, b_x, b_y \in \mathbb{R}$

*** Proceedings of the IEEE Visualization '95, pp. 135–142, IEEE Computer Society Press, 1995**

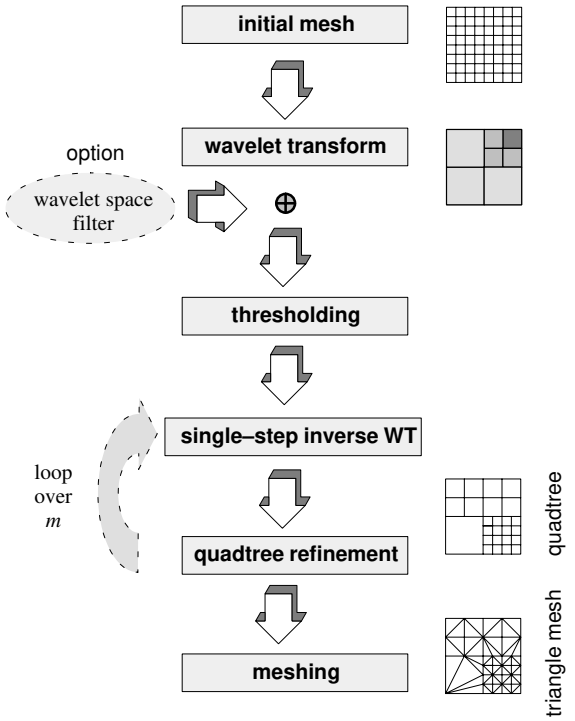


Fig. 1: Illustration of the basic concepts of the method: estimation of the surface parameters using the local detail signal of the WT, point removal and quadtree based meshing of the remaining surface points.

The basis functions are derived from each other by scaling and shifting one prototype function $\psi(x,y)$ controlled by the parameters a_x, a_y and b_x, b_y respectively [5].

$$\psi_{a,b}(x,y) = \frac{1}{\sqrt{|a_x\ a_y|}}\psi\left(\frac{x-b_x}{a_x},\frac{y-b_y}{a_y}\right) \tag{2}$$

$$\langle\psi_l\ ,\psi_k\rangle = \delta(l-k),\ \delta : \textit{Kronecker–delta–function} \tag{3}$$

Most discrete formulations of the 2D–WT comprise a tensor product extension along with a dyadic scaling of the bases with $a_x = a_y = 2$ and a unit shift $b_x = b_y = 1$, by which the respective bases are derived:

$$\begin{aligned}
\phi^2_{mpq}(x,y) &:= 2^{-m}\phi(2^{-m}x-p)\phi(2^{-m}y-q)\\
\psi^{2,1}_{mpq}(x,y) &:= 2^{-m}\psi(2^{-m}x-p)\phi(2^{-m}y-q)\\
\psi^{2,2}_{mpq}(x,y) &:= 2^{-m}\phi(2^{-m}x-p)\psi(2^{-m}y-q)\\
\psi^{2,3}_{mpq}(x,y) &:= 2^{-m}\psi(2^{-m}x-p)\psi(2^{-m}y-q)
\end{aligned} \tag{4}$$

$m : 1,\ldots,M$ iteration step

Consequently, any finite energy function $f(x,y) \in L^2(\mathbb{R}^2)$ can be approximated by the bases elucidated above.

$$f(x,y) = \sum_p\sum_q\left(c^M_{pq}\phi^2_{Mpq} + \sum_{m=1}^{M-1}\left(c^{m,1}_{pq}\psi^{2,1}_{mpq} + c^{m,2}_{pq}\psi^{2,2}_{mpq} + c^{m,3}_{pq}\psi^{2,3}_{mpq}\right)\right) \tag{5}$$

Note, that the previous equation provides a multiresolution hierarchy enabling the control of the bounds of any approximation. For convenience, we will denote the coefficients simply with $\overline{c_i}$.

2.2. Biorthogonal Wavelets

The final design of the wavelet bases is usually figured out by further constraining the function's shape and mathematical properties. In most computer graphics applications [10] and [7] we require strict local support along with an appropriately smooth shape, symmetry and fast decay in frequency domain. Unfortunately, these competing properties cannot be satisfied with orthonormal wavelets. Chui [3] and Unser [19], however, independently developed a class of B–spline wavelets which meet the upper requirements. The bases are not orthogonal to each other, but it is possible to set up a so–called dual frame to perfectly reconstruct the signal from the transform.

Specifically, besides of scaling function ϕ and wavelet ψ the entire transform is defined by a dual scaling function $\tilde{\phi}$ and a dual wavelet $\tilde{\psi}$.

The biorthogonal B–spline bases of order j can be defined recursively and it's scaling function follows

$$\phi_j(x) := (\phi_{j-1} * \phi_1)(x) = \int_0^1 \phi_{j-1}(x-t)\ dt,\ j \geq 2. \tag{6}$$

That is, the bases are derived from each other by self–convolution of an initial basis of order 1, and:

$$\begin{aligned}
&\phi_j(x) = \frac{1}{(j-1)!}\sum_{k=0}^{j}(-1)^k\binom{j}{k}(x-k)^{j-1}_+\\
&\begin{cases} x_+ := max(0,x)\\ x^{j-1}_+ := (x_+)^{j-1}, \quad j \geq 2\end{cases}
\end{aligned} \tag{7}$$

Note, that the support of a B–spline basis is always bound by $[0,j]$. Furthermore, the scaling functions are symmetric with respect to the center of support.

The symmetry of the corresponding wavelet is restricted to an even order. Fig. 2 shows the functional course of B–spline wavelets of increasing order. The first order type is orthogonal and known as the Haar wavelet.

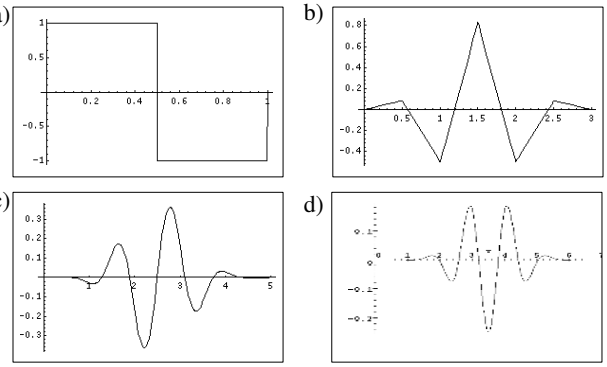


Fig. 2: Cardinal B–spline wavelets of increasing order:
a) order 1. b) order 2.
c) order 3. d) order 4.

The implementation of biorthogonal wavelet transforms employs the well known QM–Filter pyramids [19].

2.3. How to Recover Detail Signals

One problem arising with the fast QMF implementations of the wavelet transform is, that we need access to the difference signal in each iteration step m of the reconstruction. This is necessary because the detail signal at a particular mesh vertex finally decides whether or not it can be removed. For this purpose, the reconstruction pyramid has to be modified, as indicated in Fig. 3. The procedure recovers the full size detail signals $\Delta_m f$ represented by all wavelets at

$m=1,..,M$ and by the scaling functions. This can be accomplished by reversing the trace of each detail signal from the original down the decomposition pyramid. In other words, any detail signal $\Delta_m f$ at iteration depth m can be obtained from the respective wavelet coefficients by subsequent filtering and upsampling. The final output results from superimposing all detail signals:

$$f(x,y) = \sum_{m=1}^{M} \Delta_m f(x,y) \tag{8}$$

The required extensions of the QM–filterbank are straightforward.

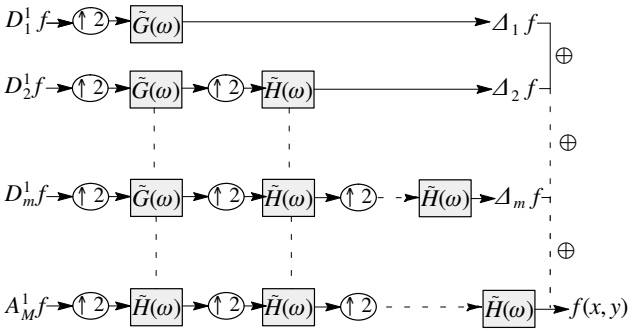


Fig. 3: Modified 1D–version of a QM–Filterbank to recover the detail signals

Note, that this procedure requires additional computation, but although the wavelet coefficients are arranged on a dyadic grid, the Heisenberg principle prevents taking them as a direct criterion for vertex removal. We will address this problem again in chapter 3.

2.4. Significance of Wavelet Coefficients

Once the data set is transformed into wavelet space, it is necessary to find appropriate criteria to control the accuracy of the surface approximation provided by the wavelet bases. Furthermore, a norm has to be found as a framework for the definition of error bounds. This can be accomplished using the signal energy E_{ges} which is defined by the L^2–norm. Hence, we filter the coefficients according to:

$$\tilde{c}_i := \begin{cases} 0, & |\bar{c}_i|^2 < \tau \\ \bar{c}_i, & |\bar{c}_i|^2 \geqq \tau \end{cases} \tag{9}$$

Increasing τ will result in increasing the error bounds of the approximation and decreasing τ will also decrease the approximation error. A canonic quantification of the error is given by the ratio of the remaining energy E_r and E_{ges}.

2.5. Level–of–detail Filtering in Wavelet Space

The introduction of an energy threshold provides a tool for globally influencing the approximation of the wavelets. However, one of the major strengths of the WT has not been harvested so far: *the localization properties*. The local support of the basis functions allows us to localize them both in spatial and in frequency domain and rejecting a particular basis will only affect its area of support. This important property enables an elegant control of the local level–of–detail of the approximation. For this purpose, the coefficients have to be weighted according to the definition of the ROI which corresponds to a filter operation in wavelet space. It can be defined in analogy with the well known filters in spatial or frequency domain. Since the filter affects the local frequency characteristics of the signal, we propose to call it ***wavelet space filter***.

Let $g(x,y)$ be a Gaussian weighting function, centered at (x_0, y_0), scaled by (σ_x, σ_y) and rotated by Θ which quantifies the level–of–detail around some location in space (x_0, y_0) and whose elliptical shape is depicted in Fig. 4a.

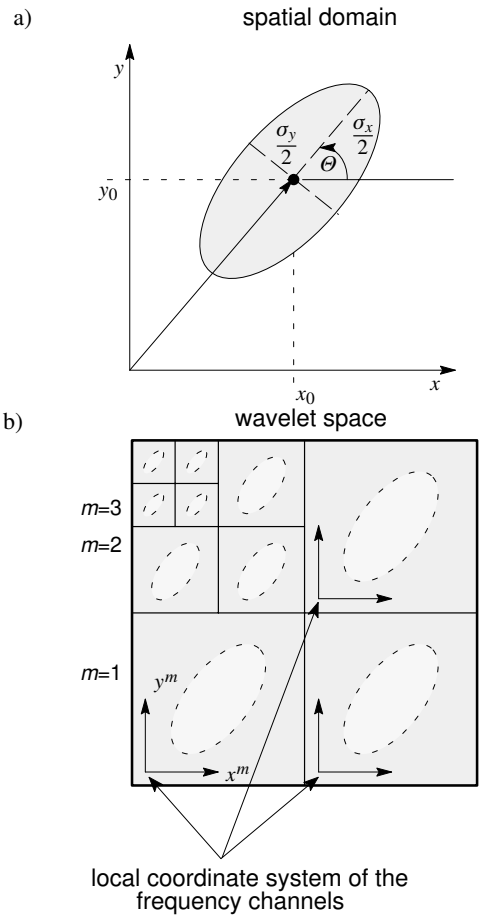


Fig. 4: Filtering in wavelet–space:
a) Rotated, translated and scaled 2D–Gaussian weighting function in spatial domain.
b) Transform of the filter into wavelet space results in multiple Gaussians located in each channel.

In order to compute its transformation into wavelet space, we have to note that any point (x_0, y_0) in spatial domain can only be located within the Heisenberg bound in wavelet domain. Furthermore, the spatial localization decreases with increasing iteration depth m.

With the dyadic scale of our 2D–WT, however, the Gaussian splits into all frequency channels and their centers are carried out in wavelet space according to:

$$x_0^m := \frac{x_0}{2^m}, \quad y_0^m := \frac{y_0}{2^m} \tag{10}$$

where x^m and y^m denote the local coordinates of the frequency channels of depth m, as presented in Fig. 4. Their variances scale according to:

$$\sigma_x^m := \frac{\sigma_x}{2^m}, \quad \sigma_y^m := \frac{\sigma_y}{2^m} \tag{11}$$

and the rotation angle Θ is invariant to the transform.

The set of Gaussian weighting functions $\{g^m(x^m, y^m)\}$ in wavelet space can be elegantly described by using homogeneous coordinates:

$$g^m(x^m, y^m) = e^{-(\boldsymbol{R}^m \cdot \boldsymbol{p}^m)^T (\boldsymbol{R}^m \cdot \boldsymbol{p}^m) + 1} \tag{12}$$

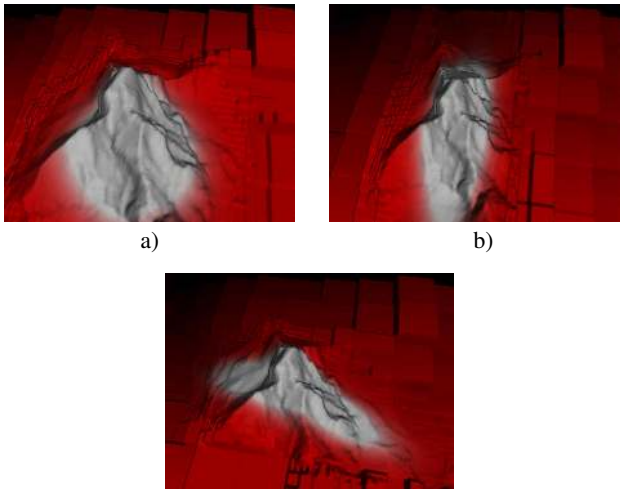
The matrix $\boldsymbol{R}^m$ stands for the affine transform of the Gaussian:

$$\boldsymbol{R}^m = \begin{pmatrix} \frac{\cos\Theta}{\sigma_x^m} & \frac{-\sin\Theta}{\sigma_x^m} & \frac{-x_0^m}{\sigma_x^m} \\ \frac{\sin\Theta}{\sigma_y^m} & \frac{\cos\Theta}{\sigma_y^m} & \frac{-y_0^m}{\sigma_y^m} \\ 0 & 0 & 1 \end{pmatrix} \tag{13}$$

and $\boldsymbol{p}^m = (x^m, y^m, 1)^T$ denotes a position in homogeneous coordinates.

To summarize this section: the control of the local level–of–detail of the wavelet approximation can be accomplished by using one single Gaussian weighting function which surrounds the region of interest in the initial data set. This Gaussian can be interpreted as a filter which is transformed into multiple Gaussians, one in each frequency channel in wavelet space. Premultiplying the coefficients with these Gaussian maps forces any subsequent thresholding to pass only coefficients located within the selected ROI. All others will be removed and hence the reconstructed signal will be most accurate within the ROI along with a Gaussian smoothing of the boundaries. Figure 5 stresses the effect of level–of–detail filtering. The digital terrain model is decomposed with Haar wavelets and filtered with Gaussians of different locations and parameters. The model is perfectly reconstructed within the focus of the Gaussian, whereas only the scaling functions represent the data outside. In the boundary region, less and less high frequency information is provided and the data becomes more and more "boxlike". Obviously, the proposed wavelet space filter acts as a *"magnifying glass"* onto the data.

We recommend applying the Gaussian filter and the thresholds only to the wavelets and keeping all coefficients of the scaling function, because they carry the DC fraction of the signal.



a) b)

c)

Fig. 5: Example for filtering in wavelet space: Decomposition of Mount Matterhorn with Haar wavelets and filtering with different Gaussian space–frequency filters.
a) σ_x=25, σ_y=25, Θ=0. b) σ_x=45.5, σ_y=14.75, Θ=0.
c) σ_x=12.5, σ_y=50, Θ=0.217.

3. Quadtree Meshing

3.1. Point Removal in Regular Triangle Meshes

So far, we elaborated some mathematical criteria for approximating a surface data set, sampled on a regular grid, using a multiresolution hierarchy. In order to build an adaptive surface triangulation, however, it is necessary to remove unimportant mesh vertices and to find a triangulation of the remaining ones. The basic criterion, by which a mesh vertex is labeled as unimportant is given by the mathematical framework of the wavelet transform. Keeping in mind that any triangulation of the surface provides a planar approximation, we only have to bound the error between the original surface function $f(x,y)$ and the bilinear interpolant provided by a triangle. Supposing furthermore that the initial data is expanded by wavelet bases, the detail signal in iteration m helps us to decide whether or not each 2^m+1th mesh vertex is necessary for the triangle approximation. First, we visit each second vertex and analyze the value of the detail signal of iteration m=1. If, let's say, the detail signal $\Delta^1 f$ in some neighborhood of vertex n is sufficiently low, then the vertex is not important and the approximation can be accomplished by a linear interpolation between vertex n–1 and n+1. This scheme can now be applied recursively by subsequent computation of the detail signals $\Delta^m f, m=1,..,M$ and by visiting all dyadic vertices at positions $n = 2^m k+1$. Once the detail signal is sufficiently small and the adjacent vertices in step m–1 are already removed, we are allowed to label the current vertex as well.

As a consequence, our procedure results in recursively building a quadtree representation of the initial mesh by removing dyadic vertices. Fig. 6 again illustrates the thinning method which finally figures out the symbolic quadtree representation of the mesh vertices depicted as an example in Fig. 7. The nodes of the quadtree contain either pointers to some child–nodes, or in case of leaves, point to the entries of a vertex list.

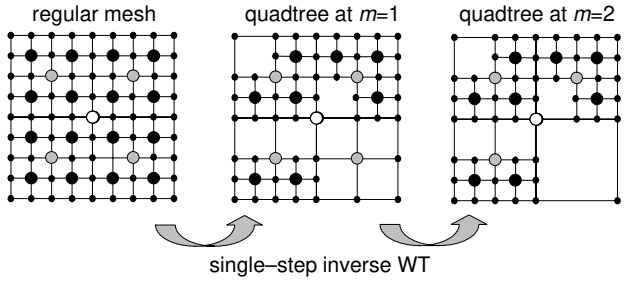


Fig. 6: Recursive growth of a quadtree from the regular mesh by analyzing the detail signals of the WT at each dyadic vertex.

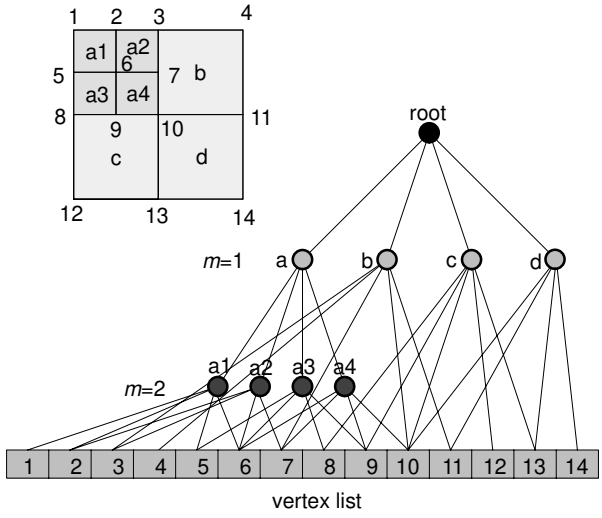


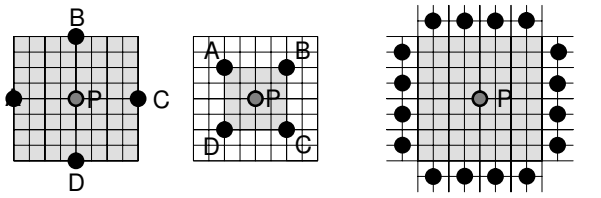
Fig. 7: Symbolic representation of the mesh using a quadtree data structure.

Note again here that the growth of our quadtree is entirely controlled by a single energy threshold τ embedded in the function

space of the wavelets. The final maximum depth of the tree depends on the upper decomposition bound M of the WT.

In order to finally decide whether or not a mesh vertex can be removed, we have to consider the following criteria which help to preserve the topology of the tree. Only in cases, where all criteria are TRUE, can the vertex be removed:

- *Wavelet–criterion*: a vertex at iteration m can be removed, if the sum of the squares of its difference signal and those within a 4–neighborhood at resolution m is less than an upper bound ε. (Fig. 8a)
- *Resolution–criterion*: a vertex at iteration m can be removed, if the four surrounding vertices at resolution m–1 were previously removed (Fig. 8b).
- *m to m–2–criterion*: a vertex can be removed, if the resulting cell is not adjacent to any cell with higher resolution than m–2. Thus, we restrict the growth to cell transitions from m to m–2 which simplifies the triangulation algorithm (Fig. 8c).



a) wavelet criterion b) resolution criterion c) m to m–2 criterion

$$\Delta_A^2 + \Delta_B^2 + \Delta_C^2 + \Delta_D^2 + \Delta_P^2 \leq \varepsilon$$

Fig. 8: Illustration of the different criteria to decide on the vertex removal.

Another aspect of the method is illustrated in Fig. 9a, where vertex P is analyzed. Suppose P survives all of the above criteria. If we remove P, however **and** if P_U and P_L are already removed, i.e. if two adjacent cells have the same resolution, then we must reject the vertices A and B on the cell boundaries, too. Hence, when traversing the vertex array from upper left to lower right, one has to keep track of upper and left vertices of the same iteration step m as well.

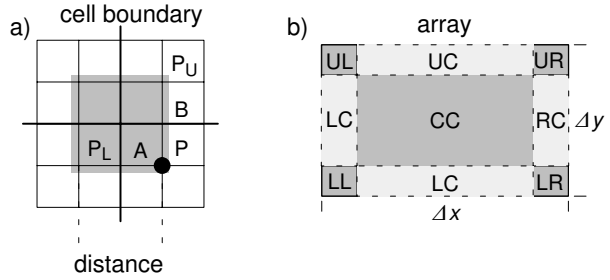


Fig. 9: a) Criteria to remove vertices on the edges of adjacent cells of the same resolution.
b) Resulting partitioning of the initial array.

This additional criterion ends up with a partitioning of the initial array into different regions (see Fig. 9b). Within these different regions, we have to check only left, upper or both adjacent vertices.

3.2. Look–up Tables for Local Triangulations

Once the tree is built from the above procedure, the quadtree cells have to be triangulated. A generic problem arising from meshing hierarchies of rectangular surface patches is the occurrence of cracks [1]. A crack occurs if we do not take care of adjacency of quadtree cells of different depth and, hence different resolution. The surface may break up, holes may appear and any consistency required for normal interpolation gets lost. Fig.10 shows a crack and also shows how to modify the triangulation to get rid of it.

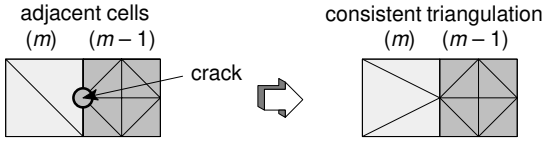


Fig. 10: The occurrence of cracks at the boundaries of adjacent quadtree cells of different resolution.

The scheme we introduce here for fast and consistent cell triangulation is based on the following observation: consider Fig. 11, where two adjacent cells are depicted along with topological arrangements that may occur for transitions from resolution m to m–1 and m–2. There are only 5 cases at the respective cell boundary. Let's presume that we restrict the growth of the quadtree so that only transitions up to m–2 are possible (*resolution criterion*). Consequently, the set of possible arrangements of vertices at the four cell boundaries can now be derived from Fig. 11. Moreover, some look–up tables may be built containing the triangulations as explained below.

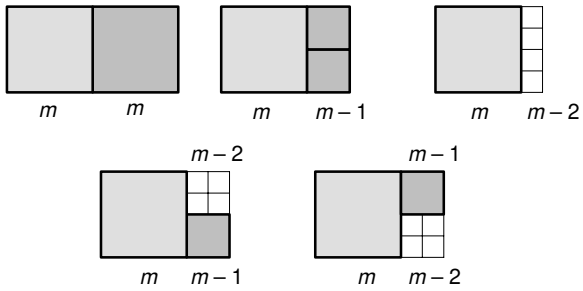


Fig. 11: Topology of mesh nodes of adjacent cells for different resolutions m, m–1 and m–2.

For cell transitions from m to m–1 a look–up table with 16 entries is built as presented in Fig. 12. The central idea of the algorithm is to first solve the triangulation within each cell for m to m–1. This is accomplished by analyzing the mesh vertices along each cell edge.

The fast computation of the look up table entry can be accomplished by a binary outcode, generated from bitwise addition of the flags of the respective edge vertices, as indicated in Fig. 12.

Once the corresponding look–up table entry is identified, we then consider mesh vertices which account for the m–2 transitions. This may cause some triangles to be split up into two pieces, as shown in Fig. 13. Consequently, the algorithm first computes the case for m to m–1 and then it decides on the corresponding subcase, by simply analyzing the flags of all intermediate vertices responsible for transitions from m to m–2. Although we get 625 possible cases, the total number of triangles required does not exceed 96. They are stored in a look–up table.

All subcases are hardcoded and contain references to these look–up table entries. Note, that although there are 625 cases only one computation of the outcode and at most 8 additional tests are necessary to compute the triangulation. It is clear that we end up with a very efficient algorithm by doing the meshing without any geometric computation but by just checking vertices along the cell edges.

A corresponding pseudocode for the recursive quadtree traversal and meshing is given with:

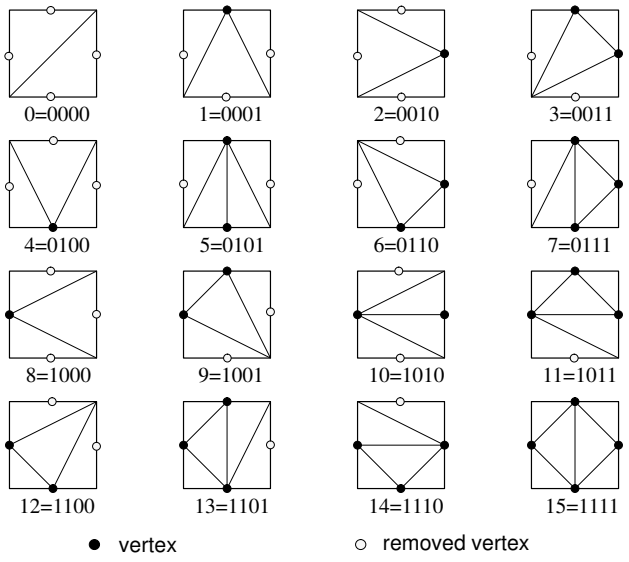


Fig. 12: Look–up table representing the optimal triangulation of all possible cases from m to m–1 and corresponding outcode.

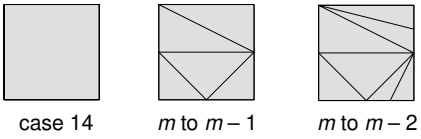


Fig. 13: Cell triangulation for cases from m to m–2 as derived from a look–up table entry of m to m–1.

```
// The initial array has a size of (N+1)(N+1).
Let N be a power of 2, N = 2^I.
Each cell is addressed by its upper left corner
vertex. //
x = y = 0; // root cell
i = I;
traverse_quadtree(x,y,i);

procedure traverse_quadtree(x,y,i)
{
      mh = 2^(i-1)    // compute center vertex of
                         son cells
      xmh = x+mh;
      ymh = y+mh;
      if (i>0) and flag(xmh,ymh)
      {
            i = i-1;
           //analyze the son cells
            traverse_quadtree(x,y,i);
            traverse_quadtree(xmh,y,i);
            traverse_quadtree(x,ymh,i);
            traverse_quadtree(xmh,ymh,i);
      }
      else
            triangulate(x,y,i);
}
```

4. Errors and Complexity

4.1. Error Analysis of Planar Approximations

One important aspect, when dealing with surface approximations is to quantify the error of the method. In our approach, error quantification is figured out by the following mean–square measure. Let $f(x,y)$ be the original surface and $g(x,y)$ be an approximation. We define the mean–square error $\overline{\Delta^2}$, as:

$$\overline{\Delta^2} = \frac{1}{(\Delta x \Delta y)} \int\int_{\Delta x \Delta y} |f(x,y) - g(x,y)|^2 \, dx\, dy \tag{14}$$

Note, that the error is normalized to the projected surface area $\Delta x \Delta y$. In the discrete case, where K samples $f_i(x_i,y_i)$ of the surface are provided at locations (x_i,y_i), i=1,..,K the mean–square error is approximated by the following relation:

$$\overline{\Delta^2} \approx \frac{1}{K} \sum_{i=1}^{K} \Delta(x_i,y_i)^2 \tag{15}$$

where, $\Delta(x_i,y_i) = f(x_i,y_i) - g(x_i,y_i)$.

Finally, in our triangle meshes the error integral of eqn. 14 is evaluated using Monte Carlo methods. For this purpose, we compute a set of K randomized locations within each of our triangles and calculate the surface value $g_i(x_i,y_i)$ by bilinear interpolation. The respective reference value for the surface $f_i(x_i,y_i)$ is obtained by bilinear interpolation of the four mesh vertices in the initial data grid as depicted in Fig. 14. The constant number of samples taken from each triangle forces the overall number of samples for the evaluation to be distributed according to the single triangle surface areas. Due to the adaptive triangulation, we end up with more samples in surface regions of high curvature and accomplish a reasonable distribution. Thus, the local mean square errors of each triangle have to be weighted with their corresponding surface area A_T^{2D} projected into 2D. The final expression of the overall mean square error of the surface yields

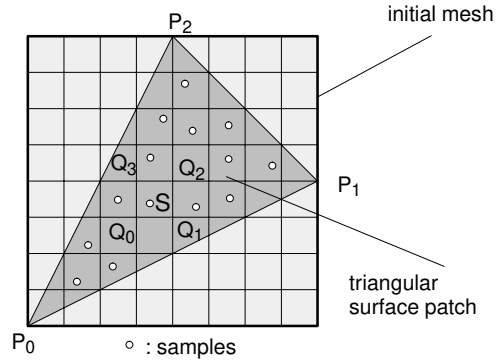


Fig. 14: Computation of the mean square error using a Monte Carlo method.

$$\overline{\Delta} = \sqrt{\frac{\sum_{\text{all triangles}} \left(\frac{1}{K} \sum_{i=1}^{K} \left(f(x_i,y_i) - g(x_i,y_i) \right)^2 A_T^{2D} \right)}{\Delta_x \, \Delta_y}} \tag{16}$$

4.2. Some remarks on Algorithmic Complexity

One of the very advantages of our method is the low algorithmic complexity for both computation of the respective transforms and for the quadtree meshing. Whereas 2D–FFT based transforms usually require $O(N^2 log_2(N))$ computations, the 2D–WT benefits from the dyadic scaling and requires only $O(N^2)$ computations. Although we have to modify the initial QMF–pyramid to compute the detail signal, the complexity still remains $O(N^2)$. The final expression for the complexity C^{WT} of a D–dimensional WT, however, depends both on the support S of the wavelet and on the iteration depth M.

$$C^{WT} = D\ N^D\ S \sum_{m=1}^{M} \left(\frac{1}{2^{D(m-1)}} \right) < \frac{2^D\ D\ N^D S}{2^D - 1} = O(N^D) \qquad (17)$$

This is another important reason for the usage of strict compact support wavelets such as the biorthogonal ones we recommend here.

Similar investigations can be carried out for the complexity of the array traversal for building up the quadtree: If the traversal is done up to the maximum depth of the WT, *M,* we have to perform at worst 4 energy tests for the wavelet criterion, 4 resolution tests and 16 tests for the *m* to *m*–2 criterion. Due to the dyadic structure of the vertices to be analyzed, we end up with

$$C^A = 24 \sum_{m=1}^{M} 2^{2I-2m} = 2^{2I}\left(8 - \frac{8}{4^M}\right) = N^2(8 - \frac{8}{4^M}) = O(N^2) \qquad (18)$$

which is still linear with respect to the overall number of mesh vertices $N^2=2^{2I}$.

5. Applications

5.1. Mesh Reduction and Error Analysis

For the following investigation, a digital terrain model of the Swiss Alps, Matterhorn/Zermatt DHM 1:25000 was selected. The initial resolution of the mesh is 256 x 256. The altitudes range from 1855.1 m (La Monta) to 4431.9 m (Matterhorn). We used cubic B–spline wavelets to decompose the data and the corresponding dual frames to approximate the reconstructions. The iteration depth was M=4, and K=3 samples were taken at each triangle to compute the mean square error. Fig. 15 illustrates, how the ratio of triangle reduction behaves as a function of the threshold τ. Furthermore, the ratio of remaining wavelet coefficients is recorded which can be interpreted as some kind of coding gain. Finally, the root of the mean–square error is plotted as well in meters. Note, that due to the logarithmic scale of the threshold, the functional behavior of both percentage of coefficients and triangles is approximately linear. The relation is further stressed in Fig. 16, where the number of triangles and the mean square error are recorded as a function of the percentage of coefficients emplyed for the approximation.

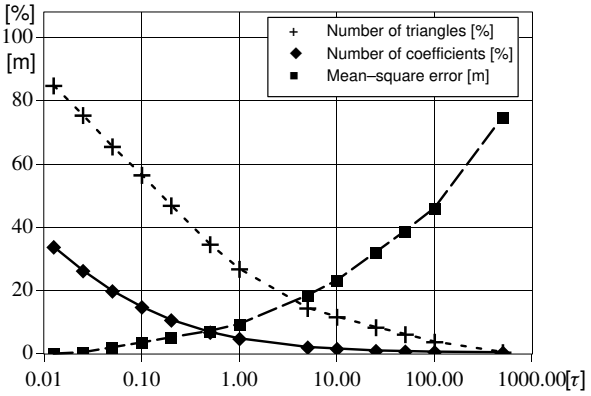


Fig. 15: Number of triangles, wavelet coefficients and mean–square error of the digital terrain model as functions of the threshold τ.

Some results of intermediate steps of the triangle reduction are depicted in Fig. 17a–c. The criteria which we defined to reject unimportant mesh vertices thin in particular in those regions of low surface curvature. This is due to the wavelet criterion which provides an estimate of the local spectral energy of the data in different frequency channels. Thus, local high frequency variations in our data force the meshing to be more dense.

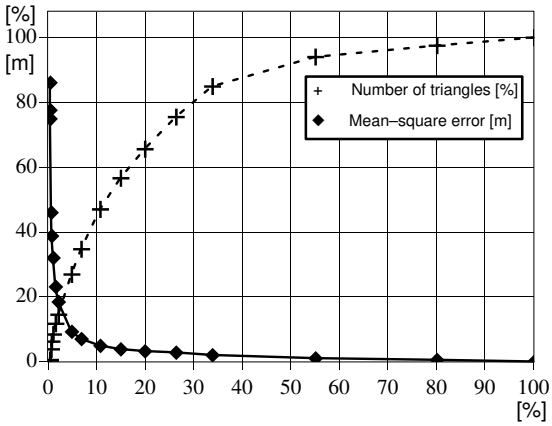


Fig. 16: Number of triangle and mean–suqare error as functions of the coefficients used for the approximation.

The corresponding Gouraud–shaded models are also presented in Fig. 17d–f where the altitude is encoded using pseudocoloring.

5.2. Level–of–Detail Control

The effect of wavelet space filtering using the Gaussian is illustrated in the images of Fig. 17. Changing the parameters of the Gaussian ellipse allows us to concentrate on the triangulation of local regions of interest. Hence, for real time animations, such as flight simulation, our method enables us to move the Gaussian for each frame according to the pilots field of vision or line of sight and to adapt the approximation to these parameters. Finally, Fig. 18d presents the Gouraud–shaded image. Obviously, the Gaussian enables the user to interact with a local "magnifying glass"

6. Conclusions

We presented a method for fast and efficient surface meshing which benefits from two basic ideas: First, any control of the surface mesh is computed by using an initial wavelet decomposition of the data samples. The mathematical framework of the WT allows us to bound the errors of the approximation and efficient criteria on whether or not single mesh vertices can be removed are provided by analyzing WT outputs. Furthermore, wavelet space filters allow a control of the quality of the surface approximation within local regions of interest and act as local "magnifying glasses". Secondly, the dyadic structure of the 2D–WT motivated us to build a quadmesh from the initial regular grid. Any triangulation of each quadtree cell is obtained by using a look–up table and hence no additional computation is required for the triangulation, as with standard Delaunay–based methods.

Due to the low complexity of this algorithm, we can achieve retriangulations of the surface at nearly interactive rates on SGI workstations. Thus, we guess that our method is particularly well suited for real–time applications, as virtual reality or flight and driving simulation. Especially, when considering low altitude flights the Gaussian filter could help to control the level–of–detail of the pilot's field of vision. Moreover, any object instance of a geometric data base related to the terrain might also be controlled by the wavelet transform. For this purpose, the actual depth of the quadtree at the object's location on the terrain is used to govern the data base and to select the object instance to be rendered.

Although the method requires an inverse WT with each new triangulation, we have proven the algorithmic complexity is still low. It is clear that we can map the WT onto special purpose hardware, such as signal processors. Currently, the method is implemented in terms of different AVS modules.

Future research has to be conducted towards extensions of the method for 3D isosurfaces in volume data using tedrahedrizations of an octree built from the WT. Additional tuning of the mesh could also be carried out by using the directional selectivity of the WT.

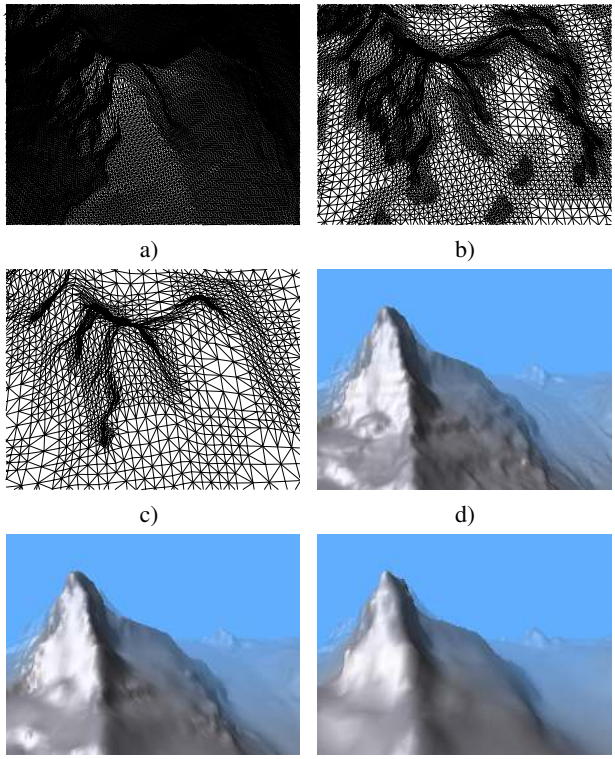


Fig. 17: Adaptive meshing of the digital terrain model. (τ: threshold, C: remaining coefficients, T: no. of triangles, $\overline{\Delta}$: mean–square error).
a)+d) τ=0.0, C=100%, T=131072, $\overline{\Delta}$=2.81.
b)+e) τ=0.5, C=6,88%, T=45106, $\overline{\Delta}$=7.02.
c)+f) τ=5.0, C=2.18%, T=19444, $\overline{\Delta}$=18.02.

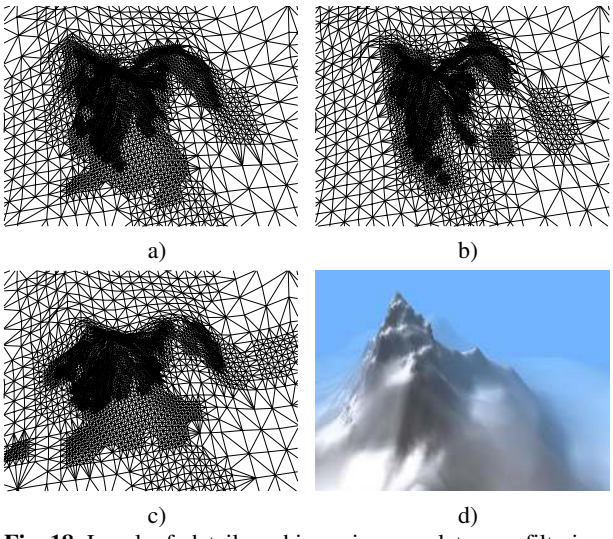


Fig. 18: Level–of–detail meshing using wavelet space filtering: (C: remaining coefficients, T: no. of triangles)
a) σ_x=20, σ_y=20, Θ=0, C=1.31%, T=9943.
b) σ_x=40, σ_y=10, Θ=0, C=1.22%, T=9236.
c) σ_x=40, σ_y=10, Θ=1.5, C=1.11%, T=9499.
d) σ_x=20, σ_y=20, Θ=0, C=1.31%, T=9943.

7. Acknowledgement

The authors like to thank the Bundesamt für Landestopographie, Bern, Switzerland for providing the digital terrain model.

8. References

[1] D. R. Baum, S. Mann and J. M. Winget, "Making Radiosity Usable: Automatic Preprocessing and Meshing Techniques for the Generation of Accurate Radiosity Solutions". *ACM Computer Graphics*, Proc. SIGGRAPH '91, pp. 51–60, 1991.

[2] J. Bloomenthal, "An Implicit Surface Polygonizer", *Graphics Gems IV*: A. Glassner, ed. Boston: Academic Press, pp. 325–349, 1994.

[3] C. Chui, *An Introduction to Wavelets*. Boston: Academic Press, 1992.

[4] A. Cohen, "Wavelets and their Applications", *Wavelets and Digital Signal Processing*. Hones and Bartlett Publishers, pp. 105–121, 1992.

[5] I. Daubechies, "The Wavelet Transform, Time–Frequency localization and signal analysis," *IEEE Trans. on Inform. Theory*, vol. 36, pp. 961 – 1005, 1990.

[6] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design - A Practical Guide.* 2nd Edition, Boston, New York: Academic Press, 1990.

[7] *Wavelets and their Applications in Computer Graphics*, A. Fournier, ed. Course Notes SIGGRAPH '94, 1994.

[8] S. J. Gortler, P. Schroeder, M. F. Cohen and P. Hanrahan, "Wavelet Radiosity", *ACM Computer Graphics, Proc. SIGGRAPH '93,* pp. 221–230, 1993.

[9] M. Gross and R. Koch, "Visualization of Multidimensional Shape and Texture Features in Laser Range Data Using Complex–Valued Gabor Wavelets", *IEEE Transactions on Visualization and Computer Graphics.* vol. 1, No. 1, pp.44–49, 1995.

[10] M. Gross, *Visual Computing.* Berlin: Springer Publishing Company, 1994.

[11] M. Gross, R. Koch, L. Lippert and A. Dreger, "A New Method to Approximate the Volume Rendering Equation using Wavelets and Piecewise Polynomials," *Computers & Graphics*, vol. 19, no. 1, pp. 47–62, 1995.

[12] T. He, S. Wang and A. Kaufman, "Wavelet–based volume morphing," *Proc. IEEE Visualization '94*, pp. 85–92, 1994.

[13] S. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, no. 7, pp. 674–693, 1989.

[14] S. R. Marschner and R. J. Lobb, "An Evaluation of Reconstruction Filters for Volume Rendering", *Proc. IEEE Visualization '94*, pp. 100–107, 1994.

[15] S. Muraki, "Volumetric shape description of range data using 'blobby model'," *Computer Graphics,* vol. 25, no. 4, pp. 227–235, 1991.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, "Wavelet Transforms", *Numerical Recipes*, Second Edition, pp. 591–606, 1992.

[17] H. Samet "The Quadtree and Related Hierarchical Data Structures". *Computing Surveys*, vol. 16, no. 2, pp. 187–260, 1984.

[18] W. J. Schroeder, J. A. Zarge and W. E. Lorensen, "Decimation of Trangle Meshes". *ACM Computer Graphics, Proc. of SIGGRAPH '92*, pp. 65–70, 1992.

[19] M. Unser, A. Aldroubi and M. Eden, "A Family of Polynomial Spline Wavelet Transforms," *Signal Processing*, vol. 30, pp. 141–162, 1993.