Multiresolution Analysis on Irregular Surface Meshes

G.P. Bonneau *

LMC-CNRS

Abstract

Wavelet-based methods have proven their efficiency for the visualization at different levels of detail, progressive transmission, and compression of large data sets. The required core of all wavelet-based methods is a hierarchy of meshes that satisfies *subdivision-connectivity*: this hierarchy has to be the result of a subdivision process starting from a base mesh. Examples include quadtree uniform 2D meshes, octree uniform 3D meshes, or 4-to-1 split triangular meshes. In particular, the necessity of subdivision-connectivity prevents the application of wavelet-based methods on irregular triangular meshes. In this paper a "wavelet-like" decomposition is introduced, that works on piecewise constant data sets over irregular triangular surface meshes. The decomposition/reconstruction algorithms are based on an extension of wavelet-theory allowing hierarchical meshes without subdivision-connectivity property. Among others, this approach has the following features:

- it allows *exact reconstruction* of the data set, even for non-regular triangulations,
- it extents previous results on Haar-wavelets over 4-to-1 split triangulations.

Keywords and Phrases: wavelets, non-regular triangulations, compression, visualization.

1 Introduction

This paper is dedicated to the multiresolution analysis of piecewise constant data sets defined on irregular triangular surface meshes. It aims to give an efficient representation of such data sets, based on decomposition and reconstruction algorithms with respect to a specific hierarchical structure.

Hierarchical structures play a crucial role since the beginning of computer graphics. More recently multiresolution analyses based on wavelet theory have started a new research domain closely related to previous works on hierarchical methods for computer graphics ([14]). Very briefly, the idea of wavelet techniques is to code a data set as a crude approximation, followed by a sequence of so-called detail coefficients that measure the error between successive finer approximations (see a survey on wavelet methods in [8]).

*LMC-CNRS, B.P. 53, F-38041 Grenoble Cedex 9, France. http://www-lmc.imag.fr/~bonneau/

Wavelet methods assume that the mesh on which the data is defined can be reached by recursive subdivision of a basic mesh. Thus every wavelet-based scheme is associated to hierarchies that have a tree structure (where every parent node is subdivided into a set of child nodes): wavelet volume visualization ([5]) is related to octree structures, wavelet radiosity ([4]) and wavelet over triangulated domains ([10, 12, 11]) are based on quadtree structures...

On the other hand, irregular triangular meshes cannot be reached by subdivision rules, therefore hierarchical structures that have been developed to handle them are more complicated than trees. These include for example hierarchical Delaunay triangulations ([9, 3]), or progressive meshes ([6, 7]).

This paper fills a gap between wavelet methods (on subdivision hierarchies) and hierarchical structures on irregular triangular meshes, for a special type of data set.

In section 2 a wavelet-like transform is introduced, that takes as input a piecewise constant function on a triangulated domain, and outputs an approximation on a simpler triangulation of the same domain, together with a set of detail coefficients encoding the error with the input function. This decomposition reduces to a traditional Haar-wavelet transform, if the input triangulation subdivides the output triangulation. In section 3 this decomposition is used locally together with a hierarchical Delaunay triangulation, in order to define a global decomposition algorithm for piecewise constant data sets on irregular surface meshes. Two types of progressive exact reconstructions are illustrated on examples. Finally some improvements of the decomposition/reconstruction algorithms are described in section 4, and illustrated on "real size" data sets (1.3M and 920K faces).

2 The triangular wavelet-like transform

In this section we first explain why wavelet theory can't be applied for the multiresolution analysis of data sets on irregular surface triangulations. Then we explain how to extent wavelet theory in order to enable the analysis of piecewise constant data sets on irregular surface triangulations.

2.1 Need for an extension of wavelet theory

There exist numerous algorithms for reducing the number of triangles in a mesh - independently of the data that is defined on this mesh. Delaunay-removal can be applied to planar or convex triangular meshes, edge-collapse to general triangular meshes. If the mesh comes from the recursive 4-split of some triangles in a base mesh, then the obvious way to simplify it, is to replace each group of four subtriangles by their parent triangle.

The common setting of these decimation algorithms is that a set of n triangles is replaced by a set of m triangles covering the same domain, with $m < n$, as shown in fig. 1.

If there exist data defined on the finest mesh, three main questions arise:

- How should we compute the data on the new simplified meshes, in order to achieve good/best approximation of the original data ?

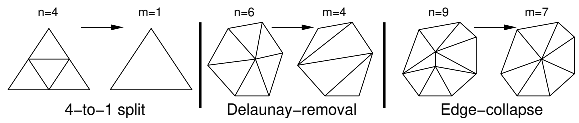

Figure 1: Local triangle decimation.

- Where should the triangle decimation be applied, in order to reach high compression ratio for a given error ?
- Is it possible to recover exactly the original data set, from any approximation ?

Since recent works ([10, 12]), it is well known that wavelet theory can answer these three questions if the triangular mesh is built by recursive 4-split. The basic fact that enables to apply wavelet theory for these special meshes is that finer triangulations subdivide coarser ones.
This is obviously not the case with Delaunay-removal or edge-collapse. Since we want to answer the three preceding questions for data sets defined on irregular surface meshes, we have to cope with the *non subdivision connectivity* of the meshes generated through the decimation algorithm.
This is possible only through an extension of wavelet theory for which the nestedness of the sequence of approximating spaces is not required anymore. This extension was introduced in a previous paper [1] in a different context. The reader should refer to [1] for a context-independent presentation of this extension.
In the following, we apply this extension for the analysis of piecewise constant data sets on irregular surface meshes.

2.2 The local decomposition and reconstruction algorithm

First some terminology: the symbol T denotes a triangle on the domain surface, s denotes a data value (a scalar) on a triangle, d denotes a detail coefficient. The superscript f (fine) denotes quantities before the local decimation algorithm, and c (coarse), after the decimation. Bold letters denote vectors. The pair $(\mathbf{T}, \mathbf{s})$ denotes the piecewise constant function equal to $\mathbf{s}_i$ on the triangles $\mathbf{T}_i$. If $\mathbf{Q}$ is a matrix, then $\mathbf{Q}_k$ denotes the k-th column vector of $\mathbf{Q}$.
We focus here on the following setting: we are given a piecewise constant function $(\mathbf{T}^f, \mathbf{s}^f)$, on n triangles, and a set of m triangles $\mathbf{T}^c$ covering the same domain on the surface, with $m < n$.

2.2.1 Analysis and synthesis formulas for irregular triangulations.

The essence of our local decomposition and reconstruction algorithm is the same as in wavelet theory: a function living in a fine space (in our case the piecewise constant function on the finer triangulation) is decomposed in a coarser approximating function (piecewise constant on the coarser triangulation) and error functions (piecewise constant on the finer triangulation). These error functions have two properties: they can be used to recover exactly the original data, and their norm is a measure of the error between the input function and the approximation.

Intuitively, we have to define one smoothing operator, that maps the input function on its approximation, and one error operator, that captures the difference between the input function and its approximation. These two operators are defined by two rectangular matrices $\mathbf{A}$ and $\mathbf{B}$ of size $m \times n$ and $(n-m) \times n$ respectively:

$$\mathbf{s}^c = \mathbf{A}\mathbf{s}^f \quad (1)$$
$$\mathbf{d} = \mathbf{B}\mathbf{s}^f, \quad (2)$$

The smoothing operator (1) computes the coarser coefficients $\mathbf{s}^c$ from the finer coefficients $\mathbf{s}^f$, and the error operator (2) computes the detail coefficients $\mathbf{d}$. The actual computation of the so-called analysis matrices $\mathbf{A}$ and $\mathbf{B}$ is detailed in the next section.

In order to keep a constant memory size for the data values, the original coefficients $\mathbf{s}^f$ are cleared from memory after the decimation, and replaced by the coarse and detail coefficients $\mathbf{s}^c$ and $\mathbf{d}$. Of course, the sum of the sizes of $\mathbf{s}^c$ and $\mathbf{d}$ equals the size of $\mathbf{s}^f$.
Since $\mathbf{s}^f$ is cleared from memory, the decomposition formulas (1) and (2) have to be invertible, in order to be able to recover the original data values. This is the purpose of the reconstruction formula:

$$\mathbf{s}^f = \mathbf{P}\mathbf{s}^c + \mathbf{Q}\mathbf{d}. \quad (3)$$

The so-called synthesis matrices $\mathbf{P}$ and $\mathbf{Q}$ are of sizes $n \times m$ and $n \times (n-m)$ respectively. Intuitively, the operator $\mathbf{P}$ is the inverse of the smoothing operator $\mathbf{A}$: $\mathbf{P}$ acts as a subdivision operator, although subdivision is not possible if the triangular domains are non-nested. The operator $\mathbf{Q}$ adds the details $\mathbf{d}$ to the oversampled data $\mathbf{P}\mathbf{s}^c$, in order to recover the original data $\mathbf{s}^f$. The matrices $\mathbf{P}$ and $\mathbf{Q}$ can be computed from $\mathbf{A}$ and $\mathbf{B}$ by:

$$\begin{pmatrix} \mathbf{P} & \mathbf{Q} \end{pmatrix} = \begin{pmatrix} \mathbf{A} \\ \mathbf{B} \end{pmatrix}^{-1}.$$

To be more precise about the properties of our decomposition, let us rewrite the reconstruction formula (3) with a functional point-of-view instead of a coefficient point-of-view:

$$(\mathbf{T}^f, \mathbf{s}^f) = (\mathbf{T}^f, \mathbf{P}\mathbf{s}^c) + \sum_{k=1}^{n-m} \mathbf{d}_k (\mathbf{T}^f, \mathbf{Q}_k). \quad (4)$$

The next remarks can be made on (4), depending on whether the fine triangulation $\mathbf{T}^f$ is a subdivision of the coarser one $\mathbf{T}^c$ or not:

- In any case, (4) is an orthogonal decomposition, and $(\mathbf{T}^c, \mathbf{s}^c)$ is always the best L^2 approximation of both $(\mathbf{T}^f, \mathbf{s}^f)$ and $(\mathbf{T}^f, \mathbf{P}\mathbf{s}^c)$.
- If $\mathbf{T}^f$ is a subdivision of $\mathbf{T}^c$, then $(\mathbf{T}^c, \mathbf{s}^c) = (\mathbf{T}^f, \mathbf{P}\mathbf{s}^c)$, and (4) can be interpreted as a Haar-wavelet decomposition. In this case, the $n-m$ functions $(\mathbf{T}^f, \mathbf{Q}_k)$ are the orthonormal Haar wavelet functions.
- On the other hand, if $\mathbf{T}^f$ is not a subdivision of $\mathbf{T}^c$, then the function $(\mathbf{T}^f, \mathbf{P}\mathbf{s}^c)$ can be interpreted as an intermediate approximation between the finer one $(\mathbf{T}^f, \mathbf{s}^f)$ and the coarser one $(\mathbf{T}^c, \mathbf{s}^c)$. This intermediate function acts as a link between two functional spaces that are not nested. And the detail coefficients $\mathbf{d}$ measure the error between this link and the finer approximation.

Figs. 2 and 3 illustrate our local decomposition on two examples. In both figures, the top part shows from left to right, the finer function $(\mathbf{T}^f, \mathbf{s}^f)$, the intermediate function $(\mathbf{T}^f, \mathbf{P}\mathbf{s}^c)$, and the final coarser function $(\mathbf{T}^c, \mathbf{s}^c)$. The bottom part shows the detail coefficients times the wavelet functions: $\mathbf{d}_k(\mathbf{T}^f, \mathbf{Q}_k)$.
Fig. 2 shows the local decomposition on a 4-to-1 split example. This leads to a traditional Haar wavelet decomposition. Note in this case that the intermediate function $(\mathbf{T}^f, \mathbf{P}\mathbf{s}^c)$ (top-middle),

although defined over a finer mesh, equals the coarser function $(\mathbf{T}^c, \mathrm{s}^c)$ (top-right).
In fig. 3, the block results from the removal of one interior vertex. Therefore, two detail coefficients are computed (bottom part).

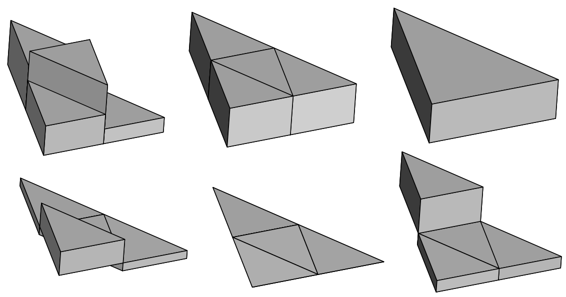

Figure 2: Local decomposition by 4-to-1 split: finer, intermediate and coarser approximations on top, detail coefficients times wavelet functions on bottom. In this case, the coarse and fine triangular domains are nested, and therefore, the intermediate approximation equals the coarser approximation. The relative high magnitudes of the detail coefficients (bottom part) show the big L^2 error between the fine and coarse approximations.

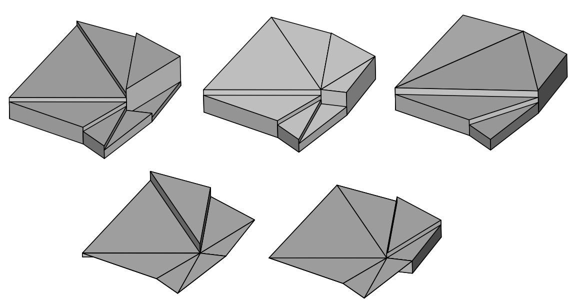

Figure 3: Local decomposition by Delaunay-removal: finer, intermediate and coarser approximations on top, detail coefficients times wavelet functions on bottom. Since the triangular domains are non-nested, the intermediate approximation differs from the coarse approximation. The relative low magnitude of the detail coefficients (bottom part) show the small L^2 error between the fine and coarse approximations.

Before going further with the computation of the analysis and synthesis matrices, we insist on two important features of our local decomposition and reconstruction:

- It can be used for every two sets of triangles $\mathbf{T}^f$, $\mathbf{T}^c$ covering the same surface domain.
- It leads to a Haar wavelet decomposition if $\mathbf{T}^f$ is a subdivision of $\mathbf{T}^c$.

2.2.2 Computation of the analysis and synthesis matrices.

Computation of A.
The analysis matrix $\mathbf{A}$ in (1) has to be chosen so that $(\mathbf{T}^c, \mathrm{s}^c)$ is the best L^2 approximation of $(\mathbf{T}^f, \mathrm{s}^f)$. This leads to the following usual choice in wavelet theory: $\mathbf{A}$ must be the matrix of the scalar products between the dual coarser scaling functions and the primal finer scaling functions. (see for ex. chap. 7.4.1. in [14] for the definition of primal and dual basis functions). In our case, the m dual coarser scaling functions equal $\frac{1}{\text{area}(\mathbf{T}_i^c)}$ on the triangle $\mathbf{T}_i^c$ and vanish elsewhere. This implies the following $m \times n$ analysis matrix $\mathbf{A}$:

$$\mathbf{A} = \left(\frac{\text{area}\left(\mathbf{T}_i^c \bigcap \mathbf{T}_j^f\right)}{\text{area}(\mathbf{T}_i^c)} \right)$$

$\mathbf{A}$ contains the relative areas of the intersections between fine and coarse triangles (see fig. 4). Therefore the value on a coarse triangle is the mean of the values on the fine triangles it intersects, weighted by the area of the intersection with each of such triangles. In the planar case, one careful way to compute $\mathbf{A}$ is to use barycentric coordinates with respect to the vertices of $\mathbf{T}_i^c$, before computing the intersections. That way, all coordinates of the intersections lie between 0 and 1, and no division has to be done to compute the relative area.

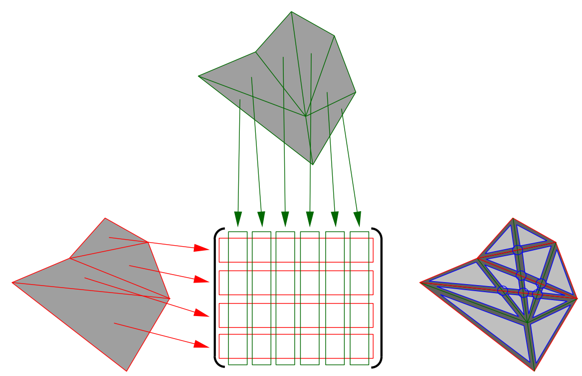

Figure 4: Computation of $\mathbf{A}$: intersections between finer and coarser triangles.

Computation of B.
To ensure the orthogonality in the decomposition (3), the functions $(\mathbf{T}^f, \mathbf{P}_j)$ must be orthogonal to $(\mathbf{T}^f, \mathbf{Q}_k)$, for all pairs (j, k), with respect to the L^2 functional scalar product. In other words, the column vectors of $\mathbf{P}$ must be orthogonal to the column vectors of $\mathbf{Q}$, for the following scalar product:

$$(\mathbf{v}, \mathbf{w}) \longrightarrow \sum_{i=1}^{n} \text{area}\left(\mathbf{T}_i^f\right) \mathbf{v}_i \mathbf{w}_i \; .$$

It can be easily shown that this is equivalent to the lines of $\mathbf{A}$ being orthogonal to the lines of $\mathbf{B}$ for the following scalar product:

$$(\mathbf{v}, \mathbf{w}) \longrightarrow \sum_{i=1}^{n} \frac{1}{\text{area}\left(\mathbf{T}_i^f\right)} \mathbf{v}_i \mathbf{w}_i \; . \tag{5}$$

This leads to an underdetermined homogeneous linear system of equations. To solve it uniquely, we fix a $(n-m) \times (n-m)$ submatrix of $\mathbf{B}$ to the identity, and we inverse the resulting linear system. This yields a first choice for the lines of $\mathbf{B}$. In a second step, we perform a Gram-Schmidt orthogonalization process (see [15], Chap. 5.2) on the lines of $\mathbf{B}$, with respect to the scalar product (5), in order to get orthonormal basis functions. Gram-schmidt orthogonalization process is known to be numerically instable for big matrices, but our matrices are always of small dimension (the dimension of our matrices is bounded by the degree of the removed vertex). Moreover orthonormality is crucial to ensure the stability of the algorithms: since the coarsest coefficients can result from millions of analysis matrices multiplications of the finest coefficients for big data sets, using non-orthonormal analysis matrices leads to instable algorithms.

To sum up, the computation of the analysis matrices $\mathbf{A}$ and $\mathbf{B}$ requires the following steps:

- computation of the intersections between coarse and fine triangles,
- computation of the relative areas $\left(\frac{\text{area}\left(\mathbf{T}_i^c \bigcap \mathbf{T}_j^f\right)}{\text{area}(\mathbf{T}_i^c)}\right)$,
- inversion of a $m \times m$ matrix,
- multiplication by a $m \times (n-m)$ matrix,
- Gram-Schmidt orthogonalization process on a $m \times n$ matrix.

3 Basic hierarchical decomposition

As previously stated, the local decomposition described in section 2 can be applied together with various triangle decimation algorithms. In this section we apply this local decomposition together with a hierarchical structure used in several previous works: the hierarchical Delaunay triangulation.
We begin by some background on hierarchical Delaunay triangulation and give the reasons for this choice. Then we show two series of examples that aim to test the stability of our method, as well as the validity of our choice for the wavelet components.

3.1 Hierarchical Delaunay triangulation.

Hierarchical Delaunay triangulations were heavily used in previous works, especially in the field of computational geometry. They were introduced in 1983 by Kirkpatrick [9] in the purpose of point location algorithms. They were later successfully applied in the field of terrain modeling ([3]).

Starting from a surface triangulation, the construction of the hierarchy can be sketched as follows:

While (# selected vertices > 0)

- select a set of at least $\frac{n}{c}$ independent (non-adjacent) vertices with degree less than d (where n is the current number of vertices, and c and d are constants: $c = 24$ and $d = 11$ were used in Kirkpatrick's paper),
- remove each of the selected vertex and their adjacent edges $\Rightarrow$ creation of blocks,
- retriangulate the blocks according to a Delaunay criterion,
- update the links between the levels.

Fig. 5 illustrates one step on the construction on a planar example: empty circles on the left denote vertices that are selected for removal. The right part shows the result after retriangulation.

Kirkpatrick proved that the hierarchy can be built in linear time, and has a logarithmic depth.
This algorithm was introduced for planar triangulations, but can be applied on more general surface geodesic triangulations, with the

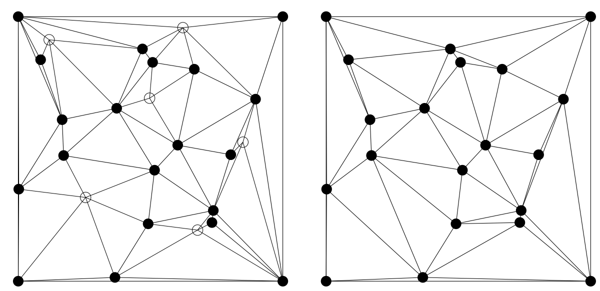

Figure 5: One step of the construction of the hierarchy: (a) independent vertices selection, (b) retriangulation.

assumption that star-shaped polygons on the surface can be triangulated. This is in particular the case for all convex surfaces. We will show planar and spherical triangulations. Our current implementation is based on a greedy algorithm in a first step, followed by a local swap of the diagonals in each quadrilateral, in order to ensure the convexity of the triangulation for spherical triangulations, or an empty circle property for planar triangulations.

This algorithm can be implemented with several data structures. Depending on the data structure, the links can join either faces, vertices or blocks between different levels. Fig. 6 shows an example of such a hierarchy. The original and simplified triangulations are shown on the left. Empty circles point to vertices that are removed to reach the next level. The middle part shows the corresponding hierarchies over the faces. Black circles denote triangles, and links join triangles that have non-empty intersections. Rounded-rectangles group triangles belonging to the same block. The right part of fig. 6 shows these blocks, and the links between intersecting blocks. Each block has two triangulations: triangles before the vertex removal (drawn solid) are called child triangles of the block, and triangles after the vertex removal (in dashed) are called parent triangles.

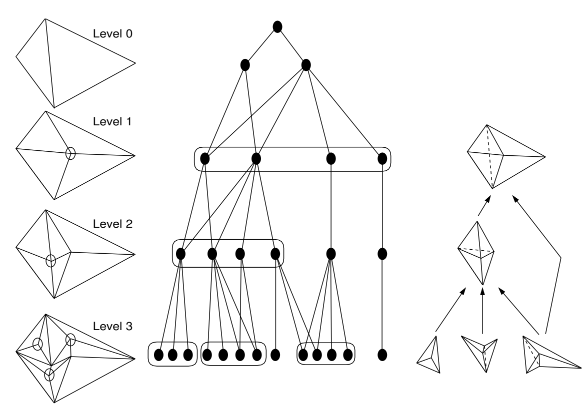

Figure 6: Hierarchical Delaunay triangulation

The main reasons for the choice of hierarchical Delaunay triangulations are the simplicity of their construction, and the linear time required for this construction. Furthermore, the analysis of the data set can be done simultaneously with the construction of the hierarchy as follows:

While (# selected vertices > 0)

- select a set of at least $\frac{n}{c}$ independent (non-adjacent) vertices with degree less than d,

- remove each of the selected vertex and their adjacent edges ⇒ creation of blocks,

- retriangulate the blocks according to a Delaunay criterion,

- **apply analysis formulas (1) and (2) for each block, in order to compute the face values on the new triangles, and the wavelet coefficients for each block,**

- update the links between the levels.

Since only one vertex is removed in each block, the number of wavelet coefficients generated in each block equals two (difference between the number of parent and child triangles in each block). Moreover, these two wavelet coefficients can be assigned to the removed vertex. This last feature will be useful in section 4, in order to improve the approximations.

After the whole analysis of the data set, each removed vertex has been assigned two wavelet coefficients, and the resulting data set is the coarsest approximation of the original one.

3.2 Level & threshold reconstructions

Once the hierarchy is constructed, and the wavelet coefficients for each block are computed, then two different kinds of partial reconstruction may be performed.

Level reconstruction
Level reconstructions result from the application of the reconstruction formula (4) on each block of the hierarchy, starting at the root level, and going down to the desired final level of reconstruction. Level reconstructions do not need the knowledge of the links between blocks. The only required information is the list of blocks, ordered from the root level to the finest level.

Threshold reconstruction
The insertion of one wavelet coefficient in the approximation implies obviously the insertion of the vertices of the block from which that wavelet coefficient has been generated. But it requires also the insertion of the blocks in the upper levels of the hierarchy, which have a link (e.g. a non-empty intersection) with that block. The necessity of including triangles that lie outside the support of the wavelet is not a consequence of using hierarchical Delaunay triangulations. In fact, the same result holds for 4-to-1 split triangulations. Fig. 7 illustrates this necessity. The top part shows the hierarchies. The solid rounded-rectangle denotes the block from which a wavelet coefficient has to be inserted. The dashed rounded-rectangles show the blocks that have to be inserted because of the hierarchy, in order to avoid hanging nodes. The bottom part of fig. 7 shows the corresponding triangulation: Solid triangles are the ones which triangulate the support of the wavelet. Dashed triangles are required in order to avoid hanging nodes.

Therefore we see that the partial reconstruction of the data set, using wavelet coefficients greater than a given threshold can be performed with the following steps:

- select the minimal sub-hierarchy that contains the wavelet coefficients greater than the given threshold

- insert the blocks in the sub-hierarchy, starting from the root level, and going down to the finest level of the selected wavelet functions.

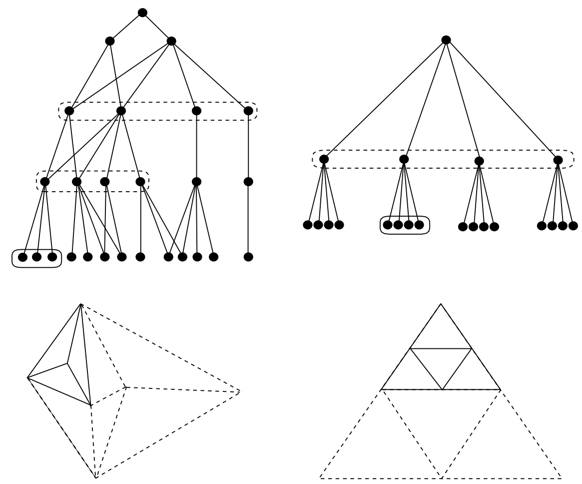

Figure 7: Including a wavelet coefficient: triangles lying outside the wavelet support are required.

3.3 Numerical results

In this section we show some numerical examples of analysis/reconstruction on a data set that was specifically constructed to test the stability of the algorithms. We have chosen a strongly irregular planar triangulation, and have mapped a non-smooth image on that triangulation: the triangulation was generated by Delaunay-inserting random points in a square domain, and the data value on each triangle equals the mean value of an image generated from the ETOPO5 data set from the National Oceanographic and Atmospheric Administration [1], showing the elevation (depth) of the earth.

The data set consists of 50002 faces. The resulting hierarchy has 32 levels. Fig. 8 shows the logarithmic plot of the number of triangles versus level index. As expected, the curve is linear, illustrating the results of Kirkpatrick ([9]). Fig. 9 shows the relative L^2-error versus number of triangles for the two partial reconstructions described in section 2.2. The approximations are clearly better by threshold reconstruction than by level reconstructions, thus illustrating the validity of our choice for the wavelet coefficients. The effect of choosing the biggest wavelet coefficients is also illustrated on fig. 10: two snapshots were taken, with almost the same number of triangles. The top part shows the reconstruction at level 21, consisting of 2326 triangles (top left=data set, top right=triangulation). The bottom part shows a threshold reconstruction consisting of 2388 triangles (bottom left=data set, bottom right=triangulation). Note how the selection of the biggest wavelet coefficients affects the triangulation, adding more triangles in discontinuous areas of the data set.

4 Improving the approximations

In this section we investigate two ways of improving the approximations resulting from the decomposition/reconstruction algorithms described in section 3. The first idea is to set some constraints during the construction of the pyramidal structure, in order to preserve important features of the data. And the second idea is to use the wavelet coefficients to construct a second pyramid that gives better approximations than the initial one. These ideas are illustrated on "real size" spherical data sets at the end of this section.

[1] Available via anonymous ftp at `ftp://ftp.ngdc.noaa.gov/Solid_Earth/Topography/tbase_5min/`

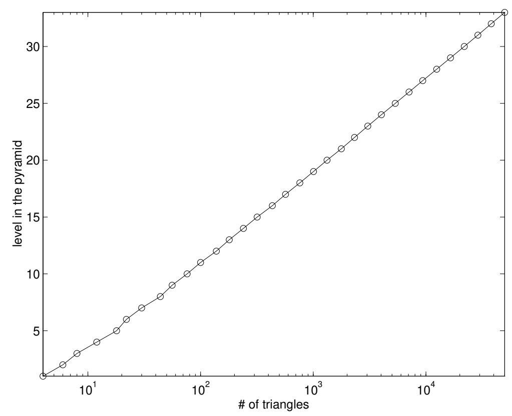

Figure 8: level vs #triangles.

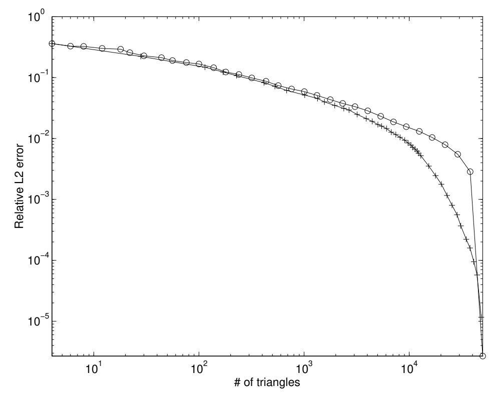

Figure 9: Relative L^2-error vs. number of triangles. 'o': level reconstruction, '+': threshold reconstruction.

4.1 Preserving important features

Using the L^2-norm as a distance error may sometimes result in the lost of important features in the data set, even for an approximation very close to the original data. Think of the coastline in an elevation/depth data set for the earth: if the coast is sharp it will be preserved in a close L^2-approximation, but if it is smooth (between England and France for example), it will be moved, or it can even disappear.

A simple yet efficient way to preserve such important features is to mark vertices, in order to forbid their removal during the construction of the hierarchal structure. We have done this for the earth elevation/depth data set (see fig. 11): all vertices along the coastlines at the finest level have been preserved during the analysis. This implies that all wavelet functions have a support that doesn't intersect the coastlines. Therefore, reconstruction with any threshold exactly reproduces the data set along the coastlines.

This approach is quite different from the one described in [13]: since their wavelet supports, defined on subdivision meshes, can intersect the coastlines, they have to check all such wavelet functions, and to require the insertion of all corresponding detail coefficients.

Our method has been used also in order to handle the data set in color plate 2: as explained in detail in section 4.3, this data set is defined on a restricted spherical domain. We were able to handle this data set by forbidding the removal of all vertices outside, and along the domain of the data set.

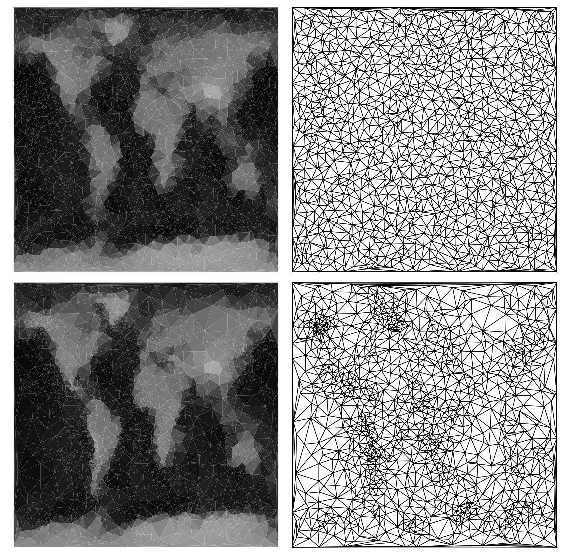

Figure 10: Top=level 21 reconstruction, 2326 triangles (left=data set, right=triangulation), Bottom=threshold reconstruction, 2388 triangles (left=data set, right=triangulation).

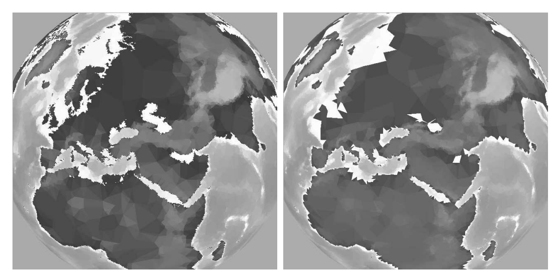

Figure 11: A coarse reconstruction (120000 faces out of 1.3M), with (left) and without (right) preservation of the coastlines. Although the error is smaller without preservation of the coastlines, the result is clearly not acceptable for this data set.

4.2 Optimizing the hierarchical structure

We already have pointed out in section 3.2 that the insertion of one wavelet coefficient implies the insertion of triangles that lie outside the support of the wavelet, in order to avoid hanging nodes. These unwanted triangles are in the upper (coarser) levels of the hierarchy, and belong to blocks that have a non-empty intersection with the wavelet support. Therefore the ideal hierarchical structure would have biggest wavelet coefficients in the upper (coarser) levels of the hierarchy, and smallest coefficients in the lower (finer) levels. Such a hierarchical structure would give optimal results, in the sense that the insertion of wavelet coefficients in decreasing order of magnitude would not require the insertion of unwanted triangles. However, constructing such an optimal hierarchy would be computationally too expensive. In order to still improve the approximations for a reasonable computational cost, we have done the following:

- compute the two wavelet coefficients for each vertex as explained in section 3,
- sort the list of removed vertices according to the sum of their squared wavelet coefficients (lowest first, biggest last),
- construct a new sequence of approximations from the original data, and removing each vertex in the order given by the previ-

ous step. Thereby two new wavelet coefficients are computed for each removed vertex.

This preprocess requires two full analysis of the data set instead of one, as well as sorting the list of vertices. On the other hand, the resulting improved sequence of approximations is easier to reconstruct: the vertices are inserted one after another together with their wavelet coefficients, regardless of the norm of these wavelet coefficients. No hierarchy has to be traversed in order to avoid hanging nodes.

Fig. 12 compares the threshold reconstruction in the numerical example of section 3.3 with the new sequence of approximations. Fig. 13 shows the resulting new approximation with the same number of triangles as in fig. 10. Note the improved concentration of the vertices in discontinuous areas of the data set.

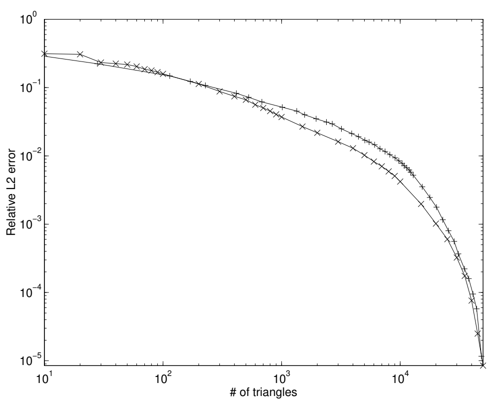

Figure 12: Relative L^2 error vs. number of triangles: '+'=threshold reconstruction with non-optimized pyramid, 'x'=reconstruction with improved sequence of approximations (see 4.2)

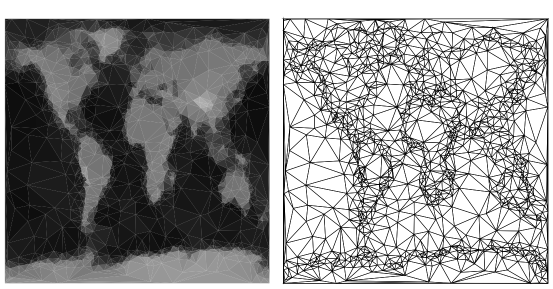

Figure 13: reconstruction with improved sequence of approximations (see 4.2), 2388 triangles (left=data set, right=triangulation).

4.3 Results

Implementation issues

In the analysis of space and time requirements of our algorithms, it is important to distinguish the geometrical part from the numerical part: the geometrical part includes computing and storing the Delaunay hierarchy of meshes, whereas the numerical part involves computing and storing the coarse and detail coefficients.

As stated in section 3.1, the geometrical part has been extensively worked in others papers. [2] for ex. describes a data structure that is based on blocks, as defined in section 3.1. The size of all proposed data structures is proportional to the number of vertices, with a bigger constant if the adjacencies are stored. The computation time for the construction of the hierarchy (without analysis of the data) is also proportional to the number of vertices. The highest cost operation is the local retriangulation of the blocks. Our current implementation, which is not optimized, works at about 3000 vertices per second on a SGI Octane 175Mhz R10000.

The numerical part of our algorithm requires the storage of the coarse and detail coefficients. As explained in section 2.2.1, this can be done with a constant memory size, since these coefficients replace the fine coefficients at each local analysis. The analysis and synthesis matrices are not stored in memory. The computation time for the numerical part is mainly due to the evaluation of the intersections between the coarse and fine triangles. Again this has not been optimized in our current code, and it works at about 4000 vertices per second on a SGI Octane 175Mhz R10000.

Examples

Color plate 1 illustrates sections 4.1 and 4.2 on a spherical data set consisting of 1.3M faces. The original data set is defined on a regular triangulation (4-to-1 split with 8 levels of subdivision starting on a icosahedron). The data value on each face equals the mean value of the ETOPO5 data set (which consists of 2160 x 4320 samples on a uniform grid). A first full analysis has been performed on the data set, with preservation of the coastlines, as explained in section 4.1. (a), (b) and (c) show three snapshots taken during this first analysis. (c) shows the coarsest level: no more vertices can be removed, the only remaining vertices are those along the coastlines, and those with degree greater than 11. Although the faces are rendered as planar, all computations are done with true spherical triangles (in particular areas and intersection computations). The vertices were sorted according to the wavelet coefficients, and a second full analysis was performed, as explained in section 4.2. (d), (e) and (f) show three snapshots taken during the reconstruction.

Color plate 2 shows the gravity anomaly on the earth. The gravity anomaly is the difference between the actual gravity, and the ideal gravity for a perfect ellipsoidal surface. The data set comes from satellite measurements. For technical reasons, it is restricted to the water surface on the earth, and between -72 and $+72$ degrees of latitude. More information on this data set, as well as the data itself are available at `http://topex.ucsd.edu/marine_grav/mar_grav.html`. The original data consists of 10800×6336 short numbers. Even numbers signify no measurement available while odd numbers contains the gravity anomaly $\times 10 + 1$.
Our method based on irregular grids can handle such complicated domains. Starting with the same regular grid as for color plate 1, we have used the method described in section 3.1 in order to remove almost all vertices on the continents and below -72 or above $+72$ degrees of latitude. Of course we did not store the hierarchy for this initial sequence of vertex removals. The resulting irregular triangular mesh consists of 920250 triangles, with small triangles covering exactly the data domain, and some few big triangles covering the areas where no data is available. Our algorithm has been applied on this initial triangulation, using the method described in section 4.1 in order to preserve the domain boundaries (i.e. coastlines and $-72^o, +72^o$ parallels). (a), (b), (c) and (d) show three snapshots taken during the reconstruction with the optimized pyramid. Deep blue indicates low gravity values, while red indicates high gravity values. (e), (f), (g) and (h) are close-up views above the pacific

ocean, where the gravity anomaly is especially irregular ((e) and (h) show the edges of the triangulation in light blue).

5 Conclusion

We have introduced algorithms for decomposing and reconstructing piecewise constant functions defined on irregular triangular meshes. Our algorithms combine hierarchical Delaunay triangulation with a wavelet-like decomposition in order to build a multiresolution model of the data set. Our local decomposition generalizes Haar-wavelets for non-nested triangulations, and is therefore especially adapted for the analysis of big and irregular data sets. Future work will address the generalization to higher order wavelets, in order to handle smoother data sets.

6 Acknowledgments

The author wishes to thank the referees for their helpful comments.

References

[1] Georges-Pierre Bonneau, Stefanie Hahmann, and Gregory M. Nielson. BLaC-Wavelets: A multiresolution analysis with non-nested spaces. In *IEEE Visualization '96*, pages 43–48. IEEE, October 1996.

[2] Mark de Berg and Katrin T. G. Dobrindt. On levels of detail in terrains. In *Technical Report UU-CS-1995-12*. Utrecht University, 1995.

[3] Leila De Floriani. A pyramidal data structure for triangle-based surface description. *IEEE Computer Graphics and Applications*, 9(2):67–78, March 1989.

[4] Steven J. Gortler, Peter Schröder, Michael F. Cohen, and Pat Hanrahan. Wavelet radiosity. In *SIGGRAPH 93 Conference Proceedings*, pages 221–230. ACM SIGGRAPH, 1993.

[5] M. H. Gross, L. Lippert, R. Dittrich, and S. Häring. Two methods for Wavelet-Based volume rendering. *Computers & Graphics*, 21(2):237–252, 1997.

[6] Hugues Hoppe. Progressive meshes. In *SIGGRAPH 96 Conference Proceedings*, pages 99–108. ACM SIGGRAPH, August 1996.

[7] Hugues Hoppe. View-dependent refinement of progressive meshes. In *SIGGRAPH 97 Conference Proceedings*, pages 189–198. ACM SIGGRAPH, August 1997.

[8] B. Jawerth and W. Sweldens. An overview of wavelet based multiresolution analyses. *SIAM Rev.*, 36(3):377–412, 1994.

[9] David Kirkpatrick. Optimal search in planar subdivisions. *SIAM Journal on Computing*, 12(1):28–35, February 1983.

[10] Michael Lounsbery, Tony D. DeRose, and Joe Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 16(1):34–73, January 1997.

[11] Gregory M. Nielson, Il-Hong Jung, and Junwon Sung. Haar-wavelets over triangular domains with applications to multiresolution models for flow over a sphere. In *IEEE Visualization '97*, pages 143–150. IEEE, November 1997.

[12] Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In *SIGGRAPH 95 Conference Proceedings*, pages 161–172. ACM SIGGRAPH, August 1995.

[13] P. Schroeder and W. Sweldens. Spherical wavelets: texture processing. In *Eurographics Rendering Workshop 1995*. Eurographics, June 1995.

[14] Eric J. Sollnitz, Tony D. DeRose, and David H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. Morgan Kaufmann Publishers, 1996.

[15] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge press, 1986.