Wavelets for Multi-resolution Analysis of Triangular Surface Meshes

Richard Southern, Patrick Marais, Edwin Blake

CS00-11-00

Collaborative Visual Computing Laboratory
Department of Computer Science
University of Cape Town
Private Bag, RONDEBOSCH
7701 South Africa
e-mail: {rsouther,patrick,edwin}@cs.uct.ac.za

Abstract

The application of Wavelets to the Multi-resolution Analysis of surfaces provides an elegant, mathematically rigorous framework for the implementation of subdivision surfaces. We present a method similar to [Lou95] for multiresolution analysis of surfaces with subdivision connectivity. However, due to error incurred during surface remeshing (as with [EDD+95, LSS+98]) we find Wavelets an unsuitable technique for feature preservation during surface compression.

1 Theory

As in Lounsbery et. al [Lou95], the technique implemented makes use of a semi-regular multi-resolution framework with biorthogonal surface wavelets. The underlying theory of wavelets and multi-resolution analysis will not be detailed here. For more information regarding terminology and theory the reader is referred to [Mal89, Dau92].

The underlying algorithm is relatively simple: Given a base-mesh / final-mesh pair (M^0, M^n respectively), the algorithm generates n hierarchical levels of resolution such that $M^0 \subset M^1 \subset \cdots \subset M^{j-1} \subset M^n$. Note that in order to generate mesh M^j each triangle $t \in M^{j-1}$ is split into 4 by introducing new points at the midpoints of each of the edges of t (as in Figure 1).

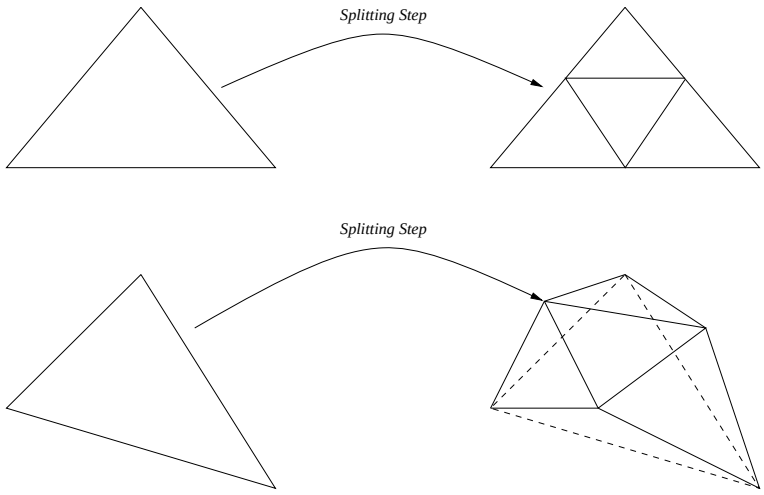

Figure 1: Examples of quadrisection. Vertices at consecutive levels introduce vertices at the midpoints of the edges at previous levels.

1.1 Refinable Basis Functions

We define $\phi_i^j(\mathbf{x})$ to be the ith scaling function at resolution j, while $\mathbf{x}$ represents a point over the domain. $\mathbf{\Phi^j}(\mathbf{x})$ is hence defined as the matrix consisting of the i functions $\phi_i^j(\mathbf{x})$. From previous work [Lou95] these functions have been proven to be refinable, and hence $\phi_i^j(\mathbf{x})$ can be written as a linear combination of the functions $\phi_i^{j+1}(\mathbf{x})$.

We now write the matrix $\mathbf{\Phi^j x})$ as

$$\mathbf{\Phi^j}(\mathbf{x}) = \left[\mathbf{O^j}(\mathbf{x})\ \ \mathbf{N^j}(\mathbf{x})\right],$$

where $\mathbf{O^j}(\mathbf{x})$ consists of the scaling functions $\phi_i^{j+1}(\mathbf{x})$ associated with the old vertices of M^j, while $\mathbf{N^j}(\mathbf{x})$ refers to the scaling functions associated with vertices added to the last mesh.

The refinability of the scaling functions allows us to define a matrix $\mathbf{P^j}$ such that

$$\mathbf{\Phi^j}(\mathbf{x}) = \mathbf{\Phi^{j+1}}(\mathbf{x})\mathbf{P^j}. \tag{1}$$

1.2 Wavelet Construction

The biorthogonal surface wavelet construction scheme of [SDS96] uses lifting to construct wavelets which are "nearly orthogonal" to the scaling functions. The strategy employed is to construct "lazy wavelets" $\mathbf{\Psi^{j-1}_{lazy}}(\mathbf{x})$ consisting of the scaling functions associated with the midpoints of the edges of M^{j-1}.

These wavelets are far from orthogonal to the scaling functions. To make them "more orthogonal" we subtract a linear combination of nearby coarse scaling functions, and define the improved wavelet as:

$$\psi_i^{j-1}(\mathbf{x}) = \phi^{\mathbf{j}}_{\mathbf{m,i}}(\mathbf{x}) - \sum_{\mathbf{k}} \left(\mathbf{s^j_{k,i}}\phi^{\mathbf{j-1}}_{\mathbf{k}}(\mathbf{x})\right) \tag{2}$$

where k is restricted to a few values corresponding to the vertices in M^{j-1} in the neighbourhood of $\phi^j_{m,i}(\mathbf{x})$. To find the values of $s^j_{k,i}$ we take the inner product of each of the terms in 2 with the scaling function $\phi^{j-1}_{i'}(\mathbf{x})$ for all i' such that the support of $\psi_i^{j-1}(\mathbf{x})$ overlaps with $\phi^{j-1}_{i'}(\mathbf{x})$, and then set $\left\langle \psi_i^{j-1} | \phi^{j-1}_{i'} \right\rangle = 0$. This results in

$$\sum_k \left(s^j_{k,i} \left\langle \phi^{j-1}_k | \phi^{j-1}_{i'} \right\rangle\right) = \left\langle \phi^j_{m,i} | \phi^{j-1}_{i'} \right\rangle. \tag{3}$$

It is convenient to write the inner product of $\langle f | g \rangle$ in matrix form. Due to the bilinearity of the inner product:

$$\langle f | g \rangle = \mathbf{g^T I^j f}$$

where **f** and **g** are column matrices consisting of the coefficients of *f* and *g* respectively, and $\mathbf{I^j}$ is the square matrix whose i, i'-th entry is $(\mathbf{I^j})_{\mathbf{i,i'}} = \left\langle \phi^{\mathbf{j}}_{\mathbf{i}} | \phi^{\mathbf{j}}_{\mathbf{i'}} \right\rangle$.

It can be shown using equation 1 that the following recurrence relation exists:

$$\mathbf{I^j} = (\mathbf{P^j})^{\mathbf{T}} \mathbf{I^{j+1}} \mathbf{P^j}. \tag{4}$$

We write the equation of 3 in matrix form, and using the recurrence relationship described in 4 we derive

$$\left[\left\langle \mathbf{\Phi^j} | \mathbf{\Phi^j} \right\rangle\right] \mathbf{S^j} = (\mathbf{P^j})^{\mathbf{T}} \left[\left\langle \mathbf{\Phi^{j+1}} | \mathbf{N^{j+1}} \right\rangle\right] \tag{5}$$

where $\left[\left\langle \mathbf{\Phi^j} | \mathbf{\Phi^j} \right\rangle\right]$ is simply $\mathbf{I^j}$, $\mathbf{S^j}$ is the matrix of the coefficients $s^j_{k,i}$ and $\left[\left\langle \mathbf{\Phi^{j+1}} | \mathbf{N^{j+1}} \right\rangle\right]$ is the submatrix of $\mathbf{I^{j+1}}$ that consists only of the columns that correspond to the members of N^{j+1}.

1.3 A Filterbank algorithm

Let $\psi_i^j(\mathbf{x})$ denote the ith locally supported wavelet approximation, and let $\Psi^j(\mathbf{x})$ be the row matrix of these functions. Define the analysis and synthesis filters by

$$[\mathbf{\Phi^j}(\mathbf{x}) \;\; \mathbf{\Psi^j}(\mathbf{x})] = \mathbf{\Phi^{j+1}}(\mathbf{x})[\mathbf{P^j} \;\; \mathbf{Q^j}], \tag{6}$$

and

$$\begin{bmatrix} \mathbf{A^j} \\ \mathbf{B^j} \end{bmatrix} = [\mathbf{P^j} \;\; \mathbf{Q^j}]^{-1} \tag{7}$$

respectively.

For lazy wavelet construction, we simply define

$$[\mathbf{P^j_{lazy}} \;\; \mathbf{Q^j_{lazy}}] = \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & & & & \\ 0 & \ddots & \ddots & \vdots & & & & \\ \vdots & \ddots & \ddots & 0 & & & \mathbf{0} & \\ 0 & \cdots & 0 & 1 & & & & \\ \hline & & & & 1 & 0 & \cdots & 0 \\ & \mathbf{P^j_m} & & & 0 & \ddots & \ddots & \vdots \\ & & & & \vdots & \ddots & \ddots & 0 \\ & & & & 0 & \cdots & 0 & 1 \end{array}\right] \tag{8}$$

where $\mathbf{P^j_m}$ is merely the matrix of connectivity information of the new vertices at step j. In order to construct what is referred to in [SDS96] as "k-disk" wavelets the above matrices are modified by the matrix $\mathbf{S^j}$ defined in equation 5 in the following fashion:

$$[\mathbf{P^j_{kd}}|\mathbf{Q^j_{kd}}] \;\; = \;\; [\mathbf{P^j_{lazy}}|\mathbf{Q^j_{lazy}} - \mathbf{P^j_{lazy}}\mathbf{S^j}] \tag{9}$$

$$\left[\frac{\mathbf{A^j_{kd}}}{\mathbf{B^j_{kd}}}\right] \;\; = \;\; \left[\frac{\mathbf{A^j_{lazy}} + \mathbf{S^j}\mathbf{B^j_{lazy}}}{\mathbf{B^j_{lazy}}}\right] \tag{10}$$

Let $\mathbf{V^j}$ denote the column vector of vertices of M^j, and $\mathbf{W^j}$ denote the corresponding matrix of wavelet coefficients. **Analysis** can be defined by

$$\mathbf{V^j} \;\; = \;\; \mathbf{A^j}\mathbf{V^{j+1}} \tag{11}$$

$$\mathbf{W^j} \;\; = \;\; \mathbf{B^j}\mathbf{V^{j+1}} \tag{12}$$

while **synthesis** is defined by

$$\mathbf{V^{j+1}} = \mathbf{P^j}\mathbf{V^j} + \mathbf{Q^j}\mathbf{W^j}. \tag{13}$$

2 Implementation

For the implementation of this wavelet scheme, it was decided to stay as close to the theoretical basis as possible, using optimisations such as sparse matrix structures and numerical solutions where possible. It can be divided into different sections.

2.1 Inner Product Matrix Calculation

Thanks to equation 5 it is not necessary to calculate the coefficients of the matrix $\mathbf{I^{j,j+1}}$, (i.e. $\left[\left\langle \mathbf{\Phi^j} | \mathbf{\Phi^{j+1}} \right\rangle\right]$) - it is sufficient to calculate only the inner product of the function with itself. The calculation of the matrix depends on the extent of the *k-disk* required - in this implementation a 1-disk wavelet will be described. *Figure 2* shows the two dimensional inner product calculation of a 1-disk wavelet.

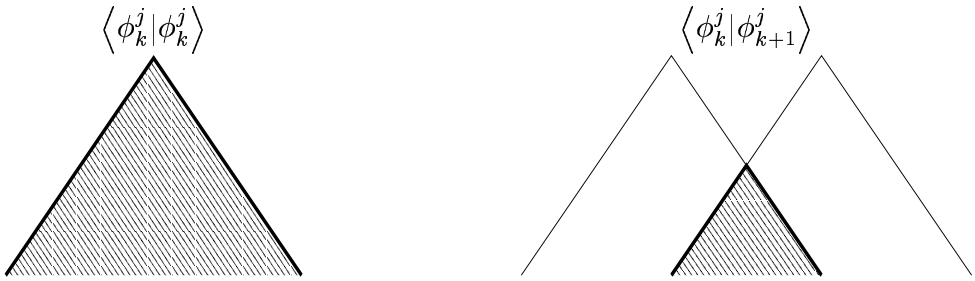

Figure 2: The inner product in 2 dimensions - the shaded region represents the overlap and hence represents the shape of the region created by the product of the two functions.

When translated into three dimensions, the problem is harder to visualise, as it does not actually correspond to logical three dimensional structures. However, the two cases of overlap of the functions are shown in Figure 4. Only a single wedge is depicted of the hat function in each case. Other than at vertices on the base mesh, there would be six such wedges surrounding each vertex - vertices on the base mesh must have fewer wedges in order to define structure. The values of the areas in each case can easily be precomputed, and numerical methods were used to generate a good approximation to the solution.

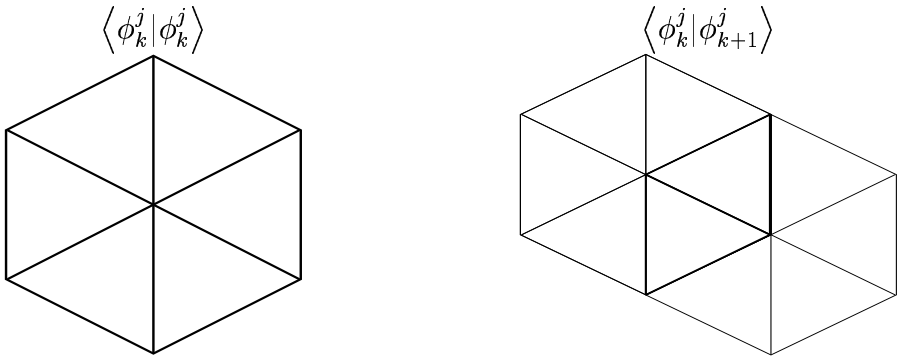

Figure 3: The two possible overlap cases of the 1-disk wavelet inner product. The highlighted region indicates the overlap of the two functions

2.2 Analysis / Synthesis Matrix Calculation

Initially it is necessary to generate the matrices $\mathbf{P^j_{lazy}}$, $\mathbf{Q^j_{lazy}}$, $\mathbf{A^j_{lazy}}$ and $\mathbf{B^j_{lazy}}$ for the desired level j from the base mesh M^0. Each of these matrices is sparse, so a sparse matrix structure was used to allow for efficient storage. These matrices are defined by Equation 8

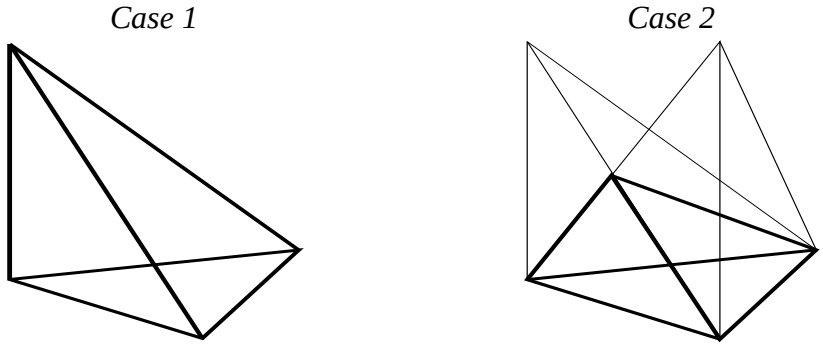

Figure 4: Two cases of overlap for individual segments of a 1-disk wavelet. *Case 1* indicates an overlap of the same function, and hence the region of the product of the two functions is represented by the tetrahedron. *Case 2* represents a case when ϕ_k^j overlaps with ϕ_{k+1}^j. Each of these cases represents a single triangle as shown in Figure 3.

and Equation 7. $\mathbf{P^1_{lazy}}$ is generated from the connectivity information of M^0, whereafter it is possible to generate the remaining $\mathbf{P^j_{lazy}}$ using only the connectivity information stored in $\mathbf{P^{j-1}_{lazy}}$.

The connectivity information stored in $\mathbf{P^j_{lazy}}$ is then used, along with the constants generated above in section 2.1 to generate the sparse matrices $\mathbf{I^j}$. It is necessary to find $\mathbf{I^{n+1}}$ for equation 5, so $\mathbf{P^{n+1}_{lazy}}$ must be found. Once $\mathbf{I^{n+1}}$ has been found the remaining $\mathbf{I^j},\ \mathbf{j = 0 \ldots n}$ are found using the recurrence relation in Equation 4. The matrix $\mathbf{S^j}$ is then determined by the formula (derived from equation 5),

$$\mathbf{S^j = (I^j)^{-1} P^j N^{j+1}}$$

where $\mathbf{N^{j+1}}$ is the submatrix of $\mathbf{I^{j+1}}$ where the columns correspond to the new vertices.

Note that at this step the inverse of the matrix $\mathbf{I^j}$ must be found - a potentially expensive operation to perform at each level j. A numerical solution was generated using Gauss-Seidel iteration. The solution is not sparse, making it extremely expensive to store. This problem can be tackled in two ways, as is described in [Lou95], but we chose merely to truncate the coefficients which were within a certain tolerance, restricting the support to be local.

The matrices $\mathbf{S^j},\ \mathbf{j = 0 \ldots n}$ are then used to calculate the matrices $\mathbf{P^j_{kd}}$, $\mathbf{Q^j_{kd}}$, $\mathbf{A^j_{kd}}$ and $\mathbf{B^j_{kd}}$ by Equations 9 and Equation 10.

2.3 Multi-resolution Analysis

Given a base mesh / final mesh pair M^0, M^n and the value n, the matrices $\mathbf{P^j_{kd}}$, $\mathbf{Q^j_{kd}}$, $\mathbf{A^j_{kd}}$ and $\mathbf{B^j_{kd}}$ are calculated as above. The matrices $\mathbf{V^j}$ and $\mathbf{W^j}$ are calculated according to equations 11 and 12.

Once all the $\mathbf{W^j}$ have been found, the $\mathbf{V^j}$ can be discarded - $\mathbf{V^0}$ is simply the vertex information of the base mesh M^0, and the remaining levels $\mathbf{V^j}$ are determined using the synthesis Equation 13.

The l_2 error incurred by each vertex is represented by the value of the detail coefficient associated with each vertex. By setting individual elements in the filter bank $\mathbf{W^j}$ to zero the vertex is removed from the reconstructed model. Compression is possible using simple stopping criteria:

- a global error tolerance ϵ could be defined indicating the sum of the detail coefficients which can be set to zero or
- the model could be compressed to a specified number of faces or vertices.

3 Implications

Wavelets for multi-resolution analysis provide a useful tool for the generation of multi-resolution models, progressive refinement and compression [SDS96] in a mathematically rigorous fashion. However, such structured solutions require equally structured input.

The calculation of matrices is a slow, mathematically intensive procedure, requiring a number of matrices to be preprocessed or processed at run-time (depending on whether time or space is to be preserved respectively). Although the computationally intensive matrix operations, such as inversion and multiplication, can be addressed in a number of ways (in this case the inversion was solved numerically, while the multiplication is a linear operation due to the implementation of a sparse matrix structure) there is a large amount of resource overhead at each level. At the very least, the matrices $\mathbf{V^j}$, $\mathbf{W^j}$, $\mathbf{P^j_{kd}}$ and $\mathbf{Q^j_{kd}}$ must be easily accessible at each level $j = 0 \ldots n$.

Subdivision connectivity (the restriction requiring all vertices to have a valence of six, except those on the base mesh) occurs very seldom in practice (except perhaps in models generated by hand). To address this issue techniques [LSS+98, EDD+95, WDSB00] can guarantee that models have this property. However these algorithms have a number of penalties.

3.1 Creating Subdivision Connectivity

Eck et. al [EDD+95] describe a strategy for generating a base mesh / final mesh pair with subdivision connectivity. They derive a base domain through a Voronoi tiling of the original mesh. Using a sequence of local harmonic maps, a parametrisation is constructed which is smooth over each triangle and is continuous at the base domain edges. The model is then re-meshed with subdivision connectivity within a certain error tolerance.

Runtime of this algorithm can be long, due to the many harmonic map computations [LSS+98]. This problem has been addressed more recently by improving the speed of these calculations, and applying heuristics to the Voronoi tile construction. However, the overall algorithm is fragile, and there is no explicit control over the number of triangles in the base domain (i.e. if the base domain patch is small, there will be an explosion of triangles within that patch at a high level of subdivision).

The re-meshing process also introduces error into the model, which can affect the highest level of detail [HG97] (if error is introduced at a low level of resolution, then the all subsequent levels of resolution will be affected). Another feature is that the topology

must remain fixed at all levels of detail, introducing the possibility for a high variation in the sizes of patches on the base mesh M^0, resulting in the triangle explosion on higher levels of subdivision, described above.

Lee et. al [LSS+98] perform a fine to coarse *decimation* of the input model to derive the base mesh. Conformal maps are used during the coarsification to immediately produce a global parameterisation of the original mesh. A hierarchical Loop smoothing filter [Loo87] is used to ensure a smooth reconstruction.

This strategy addresses most of the issues raised above - using decimation to generate the base mesh M^0 is an operation with considerably less complexity than the harmonic maps and Voronoi diagrams used above. The Loop smoothed parameterisation also ensures that the triangles on the base mesh are all similar in size, preventing a complexity explosion at higher levels of subdivision.

This technique does however require a certain measure of user control. The user is able to constrain the parameterisation to align with selected features by applying vertex and edge tags, but is not required to specify the entire patch network. Also, the mapping used to recreate the points on the final mesh may not always be a convex region, so it is possible that triangles end up on top of each other, or facing the wrong direction.

More recently, Wood et. al [WDSB00] introduce a technique of semi-regular mesh extraction from volume data sets. Assuming that data is in volume form, this would produce surfaces with subdivision connectivity without requiring a post-process of a mesh extracted with the marching cubes algorithm.

3.2 Improvements

Of importance is the preservation of particular detail features on the surface. The effectiveness of describing a surface by a wavelet scheme could be determined by the magnitude of the wavelet coefficients in the resulting filter bank $\mathbf{W}^j$. A wavelet scheme may not efficiently describe the entire model - certain regions of the filter may have small detail coefficients, while others may have large values (a spike on a flat surface is an example). This problem could be addressed in a number of ways:

- Several wavelet schemes could be used to analyse the same model - the scheme with the smallest coefficients could be used as the best representation of the surface. Given the limited number of wavelet construction strategies applicable to surfaces, there would unfortunately be a limited search space. This technique would probably not be feasible in practice, given the large amount of additional computation required.

- An adaptive, or non-linear wavelet scheme could be designed to segment the surface into separate regions, each of which could be best represented by different wavelets. This technique would incur new computational overhead due to the surface segmentation. We are not aware of any “adaptive” wavelet construction schemes at present.

4 Conclusion

This multi-resolution analysis scheme is not suitable for feature restoration. The technique incurs error and does not adequately deal with material and normal attributes.

References

[Dau92] Ingrid Daubechies. Ten lectures on wavelets. Technical report, SIAM, Philadelphia, 1992.

[EDD+95] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. *SIGGRAPH*, pages 173 – 182, 1995.

[HG97] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Technical report, Carnegie Mellon University, 1997.

[Loo87] Charles T. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, Department of Mathematics, University of Utah, 1987.

[Lou95] John Michael Lounsbery. *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*. PhD thesis, University of Washington, 1995.

[LSS+98] Aaron Lee, Wim Sweldens, Peter Schröder, Lawrence Cowsar, and David Dobkin. Maps: Multiresolution adaptive parameterization of surfaces. *SIGGRAPH*, pages 95 – 104, 1998.

[Mal89] Stephane Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989.

[SDS96] Eric J. Stollnitz, Tony Derose, and David Salesin. *Wavelets for Computer Graphics*. Morgan Kaufmann Publishers, 1996.

[WDSB00] Zoë Wood, Mathieu Desbrun, Peter Schröder, and David Breen. Semi-regular mesh extraction from volumes. *SIGGRAPH*, 2000.