EUROGRAPHICS 2005 / M. Alexa and J. Marks
(Guest Editors)

Volume 24 (2005), Number 3

GeoFilter: Geometric Selection of Mesh Filter Parameters

ByungMoon Kim Jarek Rossignac

Georgia Institute of Technology, Atlanta, USA

Abstract

When designing a lowpass filter to eliminate noise in a triangle mesh, the cutoff frequency is typically chosen by a cumbersome trial-and-error process. Therefore, it is important to provide a guideline in selecting filter frequencies. Here, we explore the relation between the frequencies in a mesh filter and the geometric measures of user-selected features. In addition, by combining previously proposed implicit and explicit formulations, we develop a second order filter that can act as lowpass, bandpass, highpass, notch, and band exaggeration/reduction filters. The proposed GeoFilter framework allows the user to choose the frequencies for that filter based on the physical size of a blob (ellipsoid) automatically fit to a user-selected feature in the mesh. For example, the size of a bump in a noisy pattern can be used as a cutoff frequency in a lowpass filter, while the size of a nose may be used to smoothen a face or to exaggerate its features as in a caricature.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Mesh Filtering

1. Introduction

The explicit formulation of a smoothing mesh filter introduced by Taubin [Tau95] has been subsequently generalized to a bandpass filter [TZG96]. Implicit forms of mesh filters followed [DMSB99] and [ZF03]. We propose a generalization of this framework that combines both explicit and implicit formulations into a more flexible second order filter. More importantly, we propose a filter, whose frequencies can be extracted automatically from the physical dimensions of a user-selected mesh feature. As illustrated in Fig. 1, the user selects a feature of the mesh (such as a nose, ear or noise bump) by spraying and diffusing paint on it. GeoFilter computes automatically an ellipsoid approximating selected feature. The dimensions of this ellipsoid guide the user in adjusting the filter parameters so as to achieve the desired result.

Filters are used profusely for audio or image signals that are regularly sampled over time or space. Since these domains are already Euclidean, regular samplings can be easily expressed in mathematical form. When constructing filters for such signals, one can directly use the property of the analog signal. For example, when the frequency greater than 10KHz needs to be attenuated, one can use the lowpass filter that has 10KHz as cutoff frequency.

In contrast, the quantitative relation between filter fre-

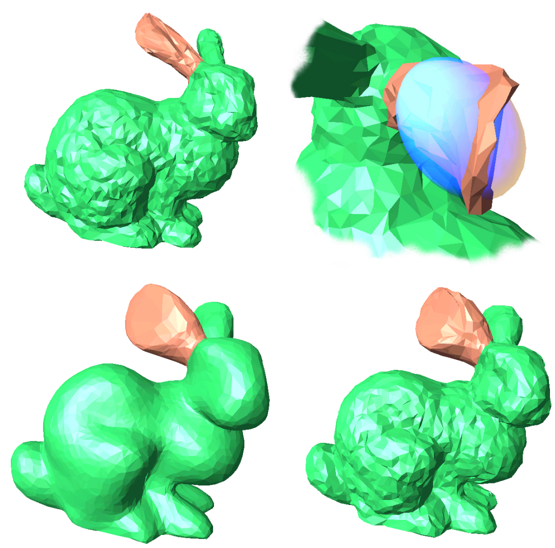

Figure 1: *The user first selects a feature (left/top). The dimensions of the feature, shown by an approximating ellipsoid (right/top), are computed, whose frequencies are used to set the filter gains. In bottom images, band exaggeration filters are applied to grow the ear, while the higher frequency bumps may be smoothened out (bottom/let) or preserved (bottom/right) by varying other filter parameters. The filter parameters are:* $s_1 = 20, s_2 = 25, G_0 = 1, G_1 = 2$ *and* $G_\infty = 0$ *(bottom/left), 0.9 (bottom/right)*

quency and size of mesh feature has been overlooked in mesh filtering. Therefore, when one wants to attenuate a noisy pattern of size 0.1, one cannot directly use this number to compute the frequency of mesh filter. In previous mesh filtering frameworks such as [Tau95, ZF03], the user must try different cutoff frequencies until the desired result is obtained. In mesh filters proposed in [DMSB99, OBB00], this quantitative relation existed but it was not explored.

Through out this paper, we assume that the vertices of the *mesh* are discrete samples on an unknown *smooth surface*. To clarify the discussion, we distinguish between *discrete operators* defined on the triangle mesh and *continuous operators* defined on the associated smooth surface.

To understand the frequencies in a mesh filter, one may consider cutting the mesh into local patches and resampling the surface along a regular grid, so as to obtain a piecewise regular domain. One may apply the sampling theorem to this regular domain together with local coordinate chart, a homeomorphism between the patch and regular domain. This way, one may analyze mesh filters mathematically. However, this approach involves non-trivial processes. Therefore, we propose to follow an alternative approach. First, we assume that the discrete triangle mesh is a close approximation of the smooth surface it samples. We further assume that the spectral properties of the smooth surface are preserved in the mesh. With this assumption, we can ignore discretization effects. We validate this idea with several examples in section 5 and Fig. 6. We show that amplification factors of our filter are very close to the ones predicted theoretically in various frequencies. We also note that the choice of discrete Laplacian operator is crucial for this assumption. We chose the formulation initially proposed in [PP93] with proper weights [OBB00, SK01] for computing the mean curvature normal vectors.

The transfer functions proposed so far in [Tau95, ZF03, DMSB99, OBB00, SK01] are limited to lowpass filtering. We propose to broaden it to other filtering applications such as exaggeration, which is explored in a multi-resolution framework in [GSS99]. To achieve this, we propose to combine explicit [Tau95, OBB00] and implicit [ZF03, DMSB99, SK01] forms together to have more flexibility in designing the mesh filter. In our construction of a second order filter, we show that the resulting framework allows band pass, notch, band exaggeration with optional high frequency reduction and high pass filters as well as lowpass filters.

2. Surface Smoothing

2.1. Choice of Discrete Laplacian Operator

The algorithms for smoothing triangle meshes have been studied extensively. In Taubin's work [Tau95], the low frequency modes in a mesh are preserved, while high frequency modes are attenuated. The underlying theory is that the eigenvector of the negative Laplacian operator of a surface represents different frequency modes of the surface, *i.e.*, the larger the associated eigenvalue is, the higher the frequency mode it represents. However, the quantitative relation between the eigenvalues of the discrete Laplacian matrix and the frequency of the noise on the surface is not known. Furthermore, the discrete Laplacian operator does not approximate the continuous Laplacian and hence one cannot expect that the eigenvalue/eigenvector pair approximates the eigenvalue/eigenfunction pair of the smooth surface. Indeed, all the eigenvalues of the discrete Laplacian operator fall into the interval $[0,2]$, whereas the eigenvalue of the smooth surface can be infinitely large. In the discrete mesh, the finer the mesh, the larger the frequency the mesh can represent and therefore eigenvalue associated with such a high frequency mode can be very large. In [ZF03], the discrete operator used for filtering is the affinity matrix, which does not have a known corresponding continuous operator for smooth mesh.

Hence, we have decided to use a discrete Laplacian operator that is convergent to the continuous one, as the mesh is refined. A popular formulation of the discrete Laplacian operator is based on the cotangent weights, originally proposed in [PP93] and used for mesh filtering in [DMSB99]. It can be shown by Taylor series expansion [Xu04] that when the cotangent weights are divided by one third of the area of neighboring triangle this formulation approximates the continuous Laplace-Beltrami operator if the mesh is regular. We show that, when this operator is used, the cutoff frequency can be selected using a more intuitive measure such as the geometric dimension of a feature.

Let $\mathbf{L} \in \mathbb{R}^{n_v \times n_v}$ be the discrete Laplacian operator defined as

$$(-\mathbf{L})_{ij} = \begin{cases} \dfrac{1}{\tilde{A}_i/3} \displaystyle\sum_{k \in \mathcal{N}_i} (\cot\alpha_{ik} + \cot\beta_{ik}) & , \quad i = j \\ -\dfrac{1}{\tilde{A}_i/3} (\cot\alpha_{ij} + \cot\beta_{ij}) & , \quad i \neq j \end{cases} \tag{1}$$

where $(\mathbf{L})_{ij}$ is the i,j-component of $\mathbf{L}$, n_v is the number of vertices, $\tilde{A}_i$ is the sum of area of triangles around i^{th} vertex, $\mathcal{N}_i$ is the set of indices of vertices around the i^{th} vertex and α_{ik}, β_{ik} are angles of the corners facing the edge connecting i^{th} and k^{th} vertices.

2.2. Decomposition of the Operator

The matrix $\mathbf{L}$ is not symmetric, but it can be decomposed into a multiplication of a diagonal matrix $\mathbf{M}$ with a symmetric matrix $\mathbf{K}$ defined as

$$\begin{aligned} -\mathbf{L} &= \mathbf{M}^{-1}\mathbf{K} \text{ , where} \\ (\mathbf{M})_{ii} &= \tilde{A}_i/3 \\ (\mathbf{K})_{ij} &= \begin{cases} \sum_{k \in \mathcal{N}_i} (\cot\alpha_{ik} + \cot\beta_{ik}) & , \quad i = j \\ -(\cot\alpha_{ij} + \cot\beta_{ij}) & , \quad i \neq j \end{cases} \end{aligned} \tag{2}$$

Note that the matrices $\mathbf{M}$ and $\mathbf{K}$ appear when one constructs a linear finite element formulation of the PDE $\dot{\phi} - \Delta\phi = 0$ of a scalar filed ϕ over the triangle mesh. The finite element formulation yields $\mathbf{M}\{\dot{\phi}\} + \mathbf{K}\{\phi\} = 0$, where $\{\phi\}$ is the long vector of ϕ sampled at the vertices, where $\mathbf{K}$ is the stiffness matrix corresponding to the negative Laplacian term, and $\mathbf{M}$ is the lumped mass matrix often preferred to the full linear formulation, not only because it gives a diagonal mass matrix, but also because it satisfies the maximum principle and hence provides stability.

Notice that if all triangles are properly oriented and there is no triangle with negative or zero area, the element stiffness matrix $\mathbf{K}_e \in \mathbb{R}^{3\times 3}$ is positive semi-definite with one dimensional null space of constant. Since $\mathbf{x}^T\mathbf{K}\mathbf{x} = \sum_{\forall e} \mathbf{x}_e^T \mathbf{K}_e \mathbf{x}_e$, where $\mathbf{x}_e \in \mathbb{R}^3$ is the collection of entries in $\mathbf{x}$ that belong to the e^{th} triangle, $\mathbf{x}^T\mathbf{K}\mathbf{x} \geq 0$ ($\mathbf{x}^T\mathbf{K}\mathbf{x} = 0$ if and only if all entries of $\mathbf{x}$ are the same when the mesh has one connected component). Consequently, $\mathbf{K}$ is (symmetric) positive semi-definite with one zero eigenvalue and $\mathbf{M}$ is a diagonal matrix with positive elements. Consider the eigen decomposition of the positive semi-definite matrix $\mathbf{M}^{-1/2}\mathbf{K}\mathbf{M}^{-1/2} = \tilde{\mathbf{V}}\Lambda\tilde{\mathbf{V}}^T$, where $\tilde{\mathbf{V}}$ is orthogonal and Λ is diagonal. By multiplying it by $\mathbf{M}^{-1/2}$ to the left and $\mathbf{M}^{1/2}$ to the right, we obtain an eigen decomposition of $-\mathbf{L}$

$$\begin{aligned} -\mathbf{L} &= \mathbf{M}^{-1/2}\left(\mathbf{M}^{-1/2}\mathbf{K}\mathbf{M}^{-1/2}\right)\mathbf{M}^{1/2} \\ &= \mathbf{M}^{-1/2}\tilde{\mathbf{V}}\ \Lambda\ \left(\mathbf{M}^{-1/2}\tilde{\mathbf{V}}\right)^{-1} = \mathbf{M}^{-1/2}\tilde{\mathbf{V}}\ \Lambda\ \tilde{\mathbf{V}}^T\mathbf{M}^{1/2} \end{aligned} \tag{3}$$

If we define $\mathbf{V} \equiv \mathbf{M}^{-1/2}\tilde{\mathbf{V}}$, then $\mathbf{V}^{-1} = \tilde{\mathbf{V}}^T\mathbf{M}^{1/2} = \mathbf{V}^T\mathbf{M}$ and we have

$$-\mathbf{L} = \mathbf{M}^{-1}\mathbf{K} = \mathbf{V}\Lambda\mathbf{V}^{-1} = \mathbf{V}\Lambda\mathbf{V}^T\mathbf{M} \tag{4}$$

Obviously, $\mathbf{V}$ is an eigenvector matrix of $-\mathbf{L}$.

2.3. Construction of Filter

Let $\mathbf{p} \in \mathbb{R}^{n_v \times 3}$ be the matrix whose columns are the x,y and z coordinates of vertices. Then, $-\mathbf{L}\mathbf{p}$ is the normal vector whose magnitude is twice the mean curvature at each vertex, and hence $-\mathbf{L}$ is the discrete Laplace-Beltrami operator [Bus92].

If we can compute $\mathbf{V}$ exactly, we can drop high frequency $\mathbf{v}_i$, yielding an ideal filtering. However, in a large mesh, it is not practical to compute more than the first few eigenvectors. Moreover, even if possible, ideal filtering often creates ripples and hence not always ideal in practice. Thus, the approach in [Tau95] is very efficient, since a carefully designed filter can keep the low frequency while reducing high frequency without having to compute the eigenvector $\mathbf{V}$.

We propose the following filtering formula inspired by the linear discrete system used in DSP and control fields.

$$\left(a_0\mathbf{I} - a_1\mathbf{L} + a_2\mathbf{L}^2 - \ldots\right)\mathbf{p}' = \left(b_0\mathbf{I} - b_1\mathbf{L} + b_2\mathbf{L}^2 - \ldots\right)\mathbf{p}$$

Applying the eigen decomposition of $-\mathbf{L}$ in (3) yields

$$(a_0\mathbf{I} + a_1\Lambda + \ldots)\mathbf{V}^{-1}\mathbf{p}' = (b_0\mathbf{I} + b_1\Lambda + \ldots)\mathbf{V}^{-1}\mathbf{p} \tag{5}$$

Let s_i be the i^{th} diagonal element of Λ. Then, the i^{th} row of (5) is

$$\mathbf{V}^{-1}\mathbf{p}' \ = \ \text{diag}\left[G(s_i)\right]\ \mathbf{V}^{-1}\mathbf{p} \tag{6}$$

where $\text{diag}\left[G(s_i)\right]$ is a diagonal matrix with $G(s_i)$ in its diagonal. The function $G(s)$ is called the transfer function and is the amplification factor for the eigenmode associated with the eigenvalue s. One can see that $G(s)$ is in the following form

$$G(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots}{a_0 + a_1 s + a_2 s^2 + a_3 s^3 + \ldots} \tag{7}$$

Finally, we have

$$\mathbf{p}' \ = \ \mathbf{V}\ \text{diag}\left[G(s_i)\right]\ \mathbf{V}^{-1}\mathbf{p} \tag{8}$$

which shows that $\mathbf{p}' = \mathbf{p}$ if $G(s) = 1$ for all s. If $G(s)$ is small (large) for some s, then $\mathbf{p}'$ will have small (large) contribution from the mode associated with it. Thus, if $-\mathbf{L}$ is the discrete Laplacian operator that approximates the continuous one, one can assume that the eigenvectors in $\mathbf{V}$ and the associated eigenvalues approximate the physical frequencies in the surface. Hence, one can design $G(s)$ to exaggerate/attenuate certain frequency pattern in the surface.

By choosing different coefficients, we can design a variety of filters, such as lowpass, high pass, bandpass, notch filters, etc. Other filters or transfer functions design methods such as classical pole-zero placements, butterworth, Chebyshev and other filters in analog/digital controls and DSP literatures can be used with slight modifications as needed.

2.4. Converting to Symmetric Matrix Equation

Since $\mathbf{L} = -\mathbf{M}^{-1}\mathbf{K}$ is not symmetric, a bi-conjugate gradient method has been used [DMSB99]. Instead, we propose to left-multiply the equation by the diagonal $\mathbf{M}$, yielding

$$\left(a_0\mathbf{M} + a_1\mathbf{K} + a_2\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \ldots\right)\mathbf{p}' = \left(b_0\mathbf{M} + b_1\mathbf{K} + \ldots\right)\mathbf{p} \tag{9}$$

Now, we have a symmetric matrix and the equation can be solved by the simpler conjugate gradient methods. We use a simple Jacobi preconditioner. Notice that the sparse matrix $\mathbf{M} + a_1\mathbf{K} + a_2\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \ldots$ does not have to be computed. The CG iteration only requires matrix vector multiplications. Thus, $\left(\mathbf{M} + a_1\mathbf{K} + a_2\mathbf{K}\mathbf{M}^{-1}\mathbf{K} + \ldots\right)\mathbf{p}$ can be conveniently computed as the sum of a cascaded series of simpler operations, such as $\mathbf{M}\mathbf{p}$, $\mathbf{K}\mathbf{p}$. The efficient computation of the Jacobi preconditioner is not trivial but it can be performed in linear time by taking advantage of the sparsity information that is available from the connectivity of the mesh.

3. Previous Works

The λ/μ filter [Tau95] is in the following form.

$$\mathbf{p}' = \left(\mathbf{I} + b_1(-\mathbf{L}) + b_2(-\mathbf{L})^2\right)^n \mathbf{p} \tag{10}$$

Thus, the transfer function is

$$G(s) = \left(1 + b_1 s + b_2 s^2\right)^n \tag{11}$$

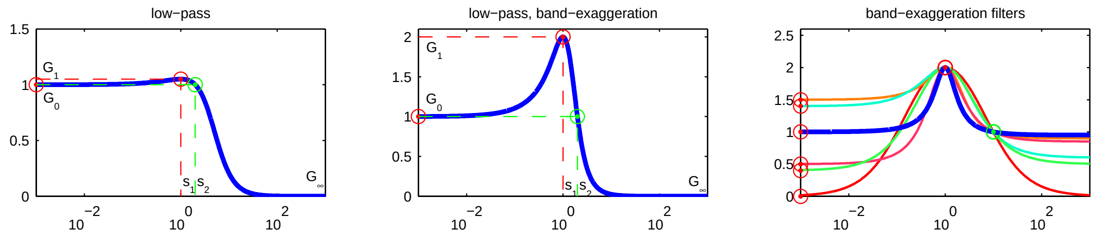

Figure 2: *Lowpass filter and its variations constructed from (15) with* $0 < s_1 < s_2, G_\infty < 1 < G_1, G_0 < G_1$

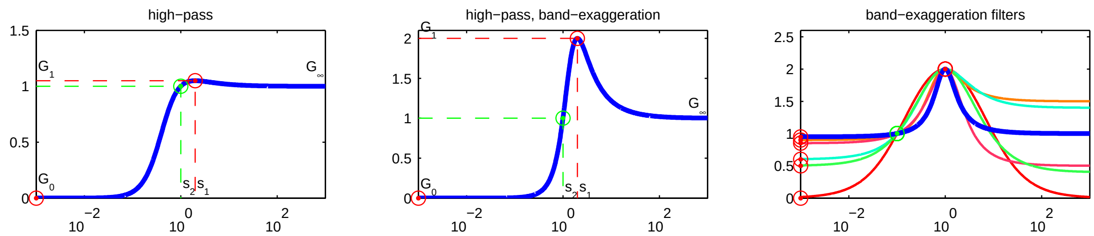

Figure 3: *Highpass filter and its variations constructed from (15) with* $0 < s_2 < s_1, G_0 < 1 < G_1, G_\infty < G_1$

Since the operator $-\mathbf{L}$ chosen in [Tau95] has eigenvalue less than two, a proper choice of b_1, b_2 will keep $G(s)$ small on the interval [0,2]. However, when one uses the Laplacian operator in (1), its eigenvalues are large, yielding large $G(s)$ for large eigenvalue s. Thus, the explicit filter formulation will not suffice for any operator that approximates the continuous Laplace-Beltrami operator. Hence, Desbrun et al used an implicit formula [DMSB99] using an operator nearly similar to (1).

$$(\mathbf{I} + a_1(-\mathbf{L}))^n \mathbf{p}' = \mathbf{p} \tag{12}$$

We can see that the transfer function is

$$G(s) = \frac{1}{(1 + a_1 s)^n} \tag{13}$$

They used a simple rescaling factor S to preserve the volume. In implicit form, $G(s)$ is safe for very large s since it will result in a very small value of G, attenuating the high frequency mode associated with it. The drawback is the lack of a flat lowpass band, which may yield an annoying shrinkage of features that one may want to preserve. Even though the operator chosen was close to (1), the quantitative meaning of the filter frequencies was not studied.

Another implicit filter is the butterworth filter found in [ZF03]. In this work, the transfer function is in the form

$$G(s) = \frac{1}{1 + a_2 s^2} \tag{14}$$

Again, the gain is a monotonically decreasing function of s and hence the attenuation of the low frequency is inevitable. A typical example is the shrinkage of the bunny ear. A higher order butterworth filter will provide more flatness at low frequencies [Cun92], since the filter is designed to have maximal flatness at zero frequency. However, to maintain a flat lowpass band, the order of the filter needs to be high and hence the cutoff rate will be very steep, approaching the ideal filtering, which can cause ripples, known as ringing. This phenomenon can happen even in the second order filter, whose maximum cutoff rate is -40db. However, in our filter construction, it can be easily reduced by choosing a larger s_2, which is defined in section 4.1.

It should be mentioned that the Laplacian operators found in [Tau95, KCVS98, KG00] regularize the mesh while performing filtering operation. The reason is complex. An intuitive understanding may be gained by deriving the Laplacian operator in [KCVS98] from the finite element framework, where edges correspond to string elements with nominal length zero, and their stiffnesses are proportional to lengths. This yields long edges pull harder and shrink, while stretching short edges, which yields mesh regularization. Unfortunately, (1) does not have a similar effect. A remedy was proposed by [OBB00]. They constructed a hybrid operator that uses (1) for normal displacement of the vertex and the umbrella operator in [KCVS98] for tangential motion with some adaptation. In contrast, our approach applies the mesh filter only once and hence the mesh regularity deteriorates little. Therefore, we do not need to embed mesh regularization.

An alternative filtering approach is to build a multiresolution hierarchy that contains different level of detail and then selectively reduce or amplify various detail levels. In [GSS99], the progressive mesh is used to build the multiresolution and then different refinement steps are zeroed, kept or amplified. Again, the mesh refinement step n and size of feature are not explicitly related and hence the user need to choose n by trial and error. Exaggeration of mesh feature can also be found in [ZG04], where they picked a feature using a geodesic fan and then searching the mesh for similar feature to exaggerate.

4. Filter Design

4.1. Exaggeration Filter Design

In this section, we explain how we designed a filter that exaggerates certain frequencies. We may also remove high frequencies and keep low frequencies, or vice versa. We use a second order polynomial for both the numerator and denominator, which gives us choices for six coefficients: $a_0, a_1, a_2, b_0, b_1, b_2$. A higher order polynomial would afford more flexibility and sharper cutoffs, but would considerably slow down computation. Therefore, in order to support interactive design, we have opted for second order polynomials.

As is shown in the far left image of Fig. 2, we allow the user to specify the DC gain $G(0) = G_0$ and the high frequency gain $G(\infty) = G_\infty$, which should be zero if one wants to attenuate the high frequencies. The user can also select the location of the maximum gain such that $G(s_1) = G_1$ to design the frequency and amplification factor. We also allow the user to specify another frequency s_2, such that $G(s_2) = 1$, to specify when the gain falls off to one. Then, given the five parameter $s_1, s_2, G_0, G_1, G_\infty$, the filter coefficients are computed as

$$
\begin{aligned}
a_0 &= 1\ ,\quad b_0 = G_0\ ,\quad a_2 = -\frac{G_0 - G_1}{G_1 - G_\infty}\frac{1}{s_1^2}\ ,\quad b_2 = a_2 G_\infty \\
a_1 &= -\frac{G_0 - G_1}{1 - G_1}\frac{2}{s_1} - \frac{1 - G_0}{1 - G_1}\frac{1}{s_2} - a_2\frac{1 - G_\infty}{1 - G_1}s_2 \\
b_1 &= \frac{G_0 - G_1}{1 - G_1}\frac{s_2 - 2s_1}{s_1^2} - \frac{1 - G_0}{1 - G_1}\frac{G_1}{s_2} - a_2 G_\infty s_2
\end{aligned}
\tag{15}
$$

where $G_1 \neq 1$.

If $s_2 > s_1 > 0, G_1 > G_0$ and $G_1 > G_\infty$, one obtain a set of filters shown in Fig. 2 that includes lowpass, band exaggeration filters with the option of high frequency reduction. In lowpass filter design, when stronger attenuation in high frequency is needed, one may choose $G_\infty = 0$. In this case, $b_2 = 0$ and the high frequency cutoff of 20db is achieved. This can be increased to 40db if $G_0 = 1$ and $s_2 = 2s_1$ since $b_1 = b_2 = 0$. The example of this steep cutoff can be found in the second image in Fig. 12, where the high frequency noise in the chin, shoulder and ear have disappeared. When we choose $G_0 = G_\infty = 0$, we obtain a bandpass filter as is shown in red in the last image of Fig. 2. If $s_1 > s_2 > 0, G_1 > G_\infty$ and $G_1 > G_0$, one obtain the highpass filter with options of various exaggerations and bandpass filter as illustrated in Fig. 3. Notice that those highpass filters will collapse the mesh since they remove low frequencies and hence are less useful than lowpass filters. However, they may still be used in some application that needs to compute the strength of high frequency signal or transfers high frequency details of a mesh to other mesh. One can also obtain a notch filter by $G_0 > 1, G_\infty > 1$ and assigning small value to G_1, as is shown in Fig. 4. An example of this notch filter can be found in the third image of Fig. 9. Also, when one increase $G_\infty > 1$ and G_1 close to one, a high frequency exaggeration filter is obtained, as is shown in Fig. 5.

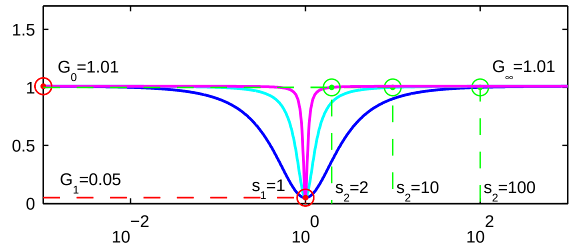

Figure 4: *Notch filters with different stop-band widths.*

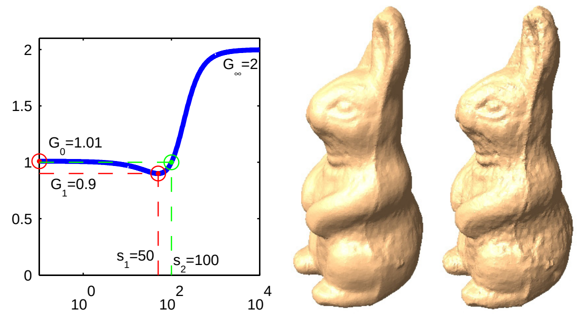

Figure 5: *High frequency amplification filter applied to a rabbit model(33,519 vertices), which took 26 seconds in 2.4GHz Pentium4.*

In conclusion, our filter formulation in (9) and (15) can be used in variety of mesh processing applications.

Note on Computation Time and Higher Order Filters

Our filter (15) is second order. The higher order filter can provide more freedom in designing $G(s)$. For example, Chebyshev filter can keep the pass band to have minimum ripple. However, as the order of the filter grows, the condition number of the matrix in the left hand side of (9) will grow exponentially and hence applying it will be very slow. [ZF03] suggest an idea that can possibly remedy this by factoring high order polynomials into a product of quadratic or linear polynomials. Zhang et al. even factored the denominator of (14) into $\left(1+\sqrt{a_2}s\right)\left(1+j\sqrt{a_2}s\right), j = \sqrt{-1}$ and then solved the two first order complex matrix equations.

Our filter process takes a few seconds for a model with 3,291 vertices. For the dinosaur model with 28,098 vertices, it requires 14∼17 seconds for high frequency filters in Fig. 9 and a few minutes for low-frequency filtering as is shown in Fig. 10. Note that low frequency filtering is much slower than high frequency one. We provide timing results for all models measured in 2.4GHz Pentium4 PC with 512MB of main memory. Notice that this computation time could be significantly improved by using complex valued conjugate gradient solver [ZF03] or the multigrid solver [NGH04].

4.2. Filters Decomposable to First Order Ones

We explore the idea of constructing a filter that can be factored into real polynomials of degree one, since such an approach allows the left hand side of (9) to be factored into

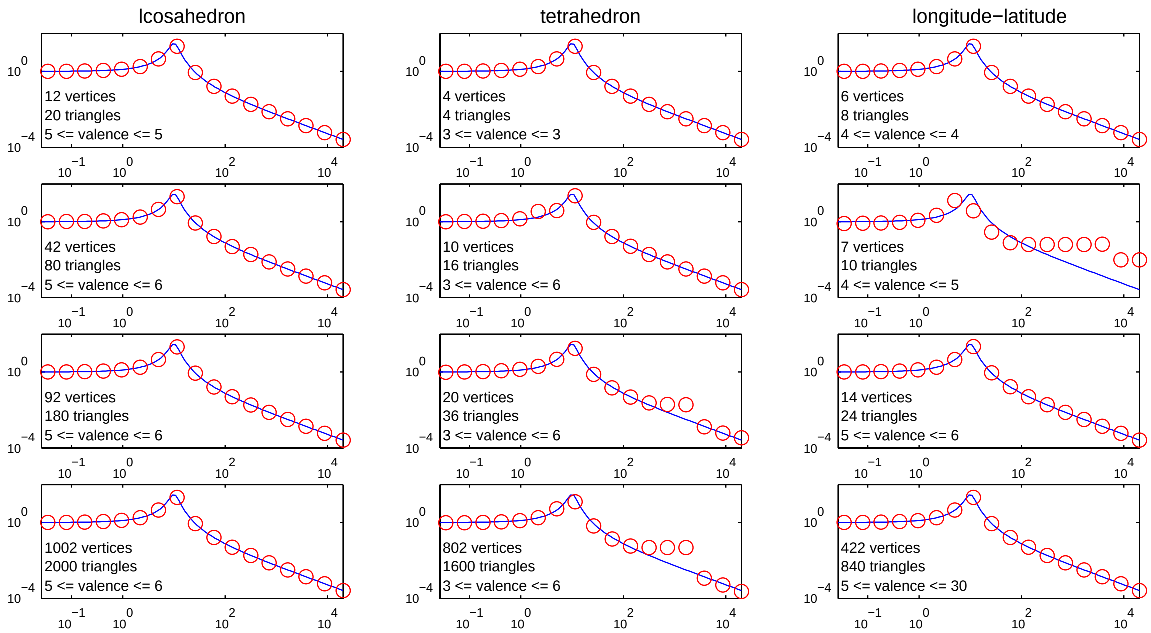

Figure 6: *Exaggeration filter($s_1 = 10$, $s_2 = 25$, $G_0 = 1.0$, $G_1 = 30$, $G_\infty = 0$) applied to sphere meshes of various radii and connectivities. The solid blue line is $G(s)$ computed from (7), while red dots are samples of experimental gains r_O/r_I, where r_I, r_O are the average radii of input and output spheres, respectively. Notice that y-axes are in log scale.*

products of $\mathbf{M}^{-1}(\mathbf{K}+p_i\mathbf{M})$, which can be solved much faster.

$$G(s) = \left(G_0 \frac{p_1 p_2 p_3 \ldots}{z_1 z_2 z_3 \ldots}\right) \frac{(s+z_1)(s+z_2)(s+z_3)\ldots}{(s+p_1)(s+p_2)(s+p_3)\ldots} \tag{16}$$

where p_i and z_i are real numbers. A classical control theory [Shi78], pages 213–226, provides an easy guideline in choosing z_i and p_i using the asymptotic lines that turns 20db at z_i and -20db at p_i as is illustrated in Fig. 8. As is shown in Fig. 7, it can be set to approximate the exaggeration filter too. However, it is difficult to make the exaggeration band narrow. In low pass filtering, it is hard to obtain a sharp cut-off rate while maintaining flat pass band. Thus, filters decomposable to first order ones can be intuitively designed by asymptotic lines and faster than the second order filters but they provide limited filtering operations.

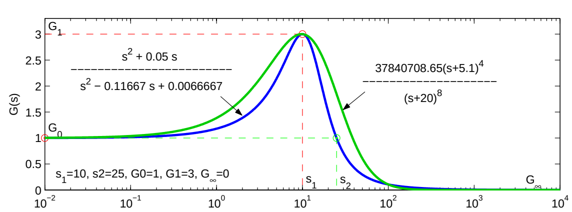

Figure 7: *Comparsion of exaggeration filter(blue) to a filter designed by asymptotic lines(green).*

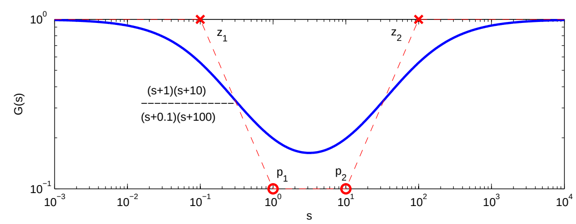

Figure 8: *Designing a band stop filter by choosing four frequencies. $G(s)$ tends to turn 20db(-20db) at each $z_i(p_i)$.*

5. Tests of Filtering Framework

We now evaluate the assumption made in section 1 under the proposed filtering framework. Since the curvature of a sphere is unique, we can predict the shrinkage ratio theoretically. We apply the filter to sphere models of various radii and then measure their shrinkage/expansion ratios and compare them to the theoretically predicted ones.

Consider a sphere equation $\mathbf{p}\cdot\mathbf{p} = r^2$. Since $-\mathbf{L}$ computes the normal vector whose length is twice the mean curvature, $-\mathbf{L}$ computes

$$-\mathbf{L}\mathbf{p} = \frac{2}{r}\frac{\mathbf{p}}{|\mathbf{p}|} = \frac{2}{r^2}\mathbf{p} \tag{17}$$

Thus, the eigenfunction is the sphere itself and the eigenvalue is $2/r^2$, which implies that if we apply the filter to spheres of radius r, it should shrink or expand by the ratio of $G\left(2/r^2\right)$.

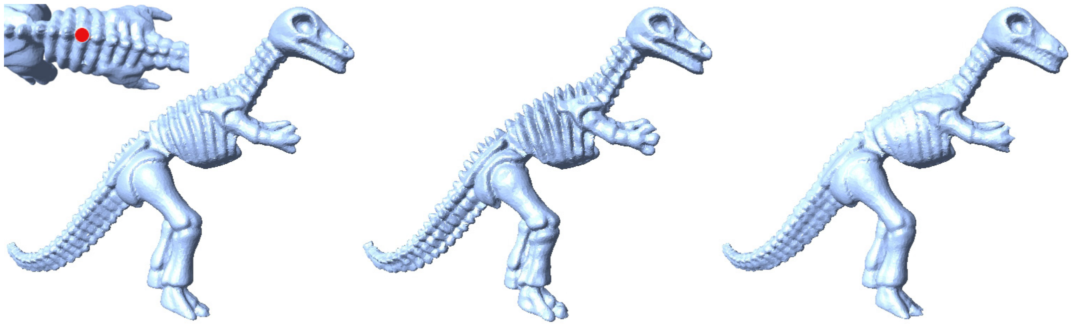

Figure 9: *Various filtering results for a dinosaur model. Far left is the initial model with 28,098 vertices and 56,192 triangles. The next two are produced by a band exaggeration and a band stop filter for $s_1 = 630$ (frequency of back-bone marked in red circle), $s_2 = 1260, G_0 = 1$ and $G_1 = 3, G_\infty = 0.9$ (second), and $G_1 = 0.01, G_\infty = 1.1(third)$. The computation times are 17 and 14 seconds, respectively.*

We apply the filters for a sphere mesh obtained from three different tessellation methods: longitude-latitude model and subdividing tetrahedron and lcosahedron. As is shown in Fig. 6, the filter output and the theoretical gain matches well in the mesh obtained from lcosahedron since the triangulation is near regular. In other models, the mesh includes vertices whose valence is significantly different from six. This leads to some imprecise results for certain frequencies. However, in most frequency, the expected frequency and filter throughput match well.

6. Selecting Feature and Computing Filter Frequency

In the proposed GeoFilter framework, the filter frequencies are chosen from the physical size of a user selected mesh feature. For example, a sphere like feature can be considered to have frequency of approximately $2/r^2$ and a cylinder like feature has $1/r^2$. Thus, if the user knows the size of the feature, the filter frequency can be easily computed.

To facilitate this process, we first allow the user to select a portion of mesh by picking a triangle graphically and then by automatically expanding the selection to neighboring triangles as long as the user keep pressing the mouse button. The operation may be repeated to extend the selection areas in a less regular fashion. The feature size is computed automatically by fitting a bounding box around optimally aligned principal axes computed as the eigenvectors of the covariance matrix [GLM96]. The bounding box is shown as a transparent ellipsoid in Fig. 1. The dimensions of the bounding box are used to derive the desired filter frequencies.

In Fig. 9, the bump of the back bone has the radius of 0.0563, which corresponds to the frequency of $2/0.0563^2 \approx 630$. This frequency is exaggerated in the center image and reduced in the right image. We apply similar strategy in various examples. In Fig 11, we pick the horse shoe that gives frequency of approximately 60. In Fig. 12, we pick nose that has frequency of approximately 25. Thus, the filter frequency can be computed from the sizes of features rather than via a time consuming trial-and-error process.

7. Results and Discussion

We apply our filters for various models in Fig. 1, 5, 9, 10, 11, and 12. All discussions as well as filter parameters and timing results on 2.4GHz Pentium4 PC are provided with figures.

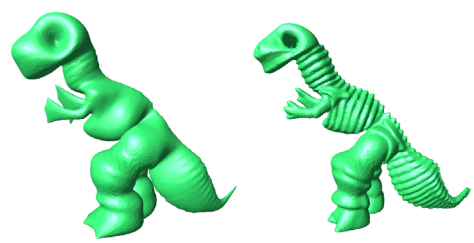

Figure 10: *A band exaggeration filter applied to the dinosaur model(far left in Fig. 9) with and without high frequency attenuation. $s_1 = 25$ is chosen as the frequency of the leg: $s_2 = 60, G_0 = 1, G_1 = 3$ $G_\infty = 0.0$ (left), and $G_\infty = 0.9(right)$. The computation times are 142 and 337 seconds, respectively*

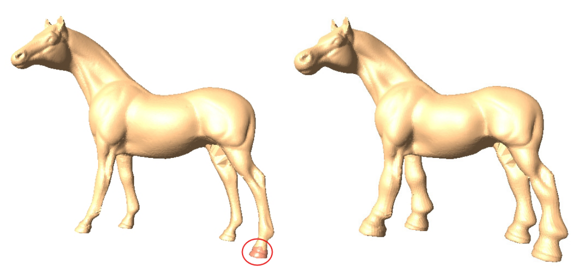

Figure 11: *Exaggeration of the legs of a horse model with 48,485 vertices. Left image is original model. Right image is obtained with exaggeration filter ($s_1 = 60, s_2 = 120, G_0 = 1, G_1 = 2, G_\infty = 0.9, 182sec.$).*

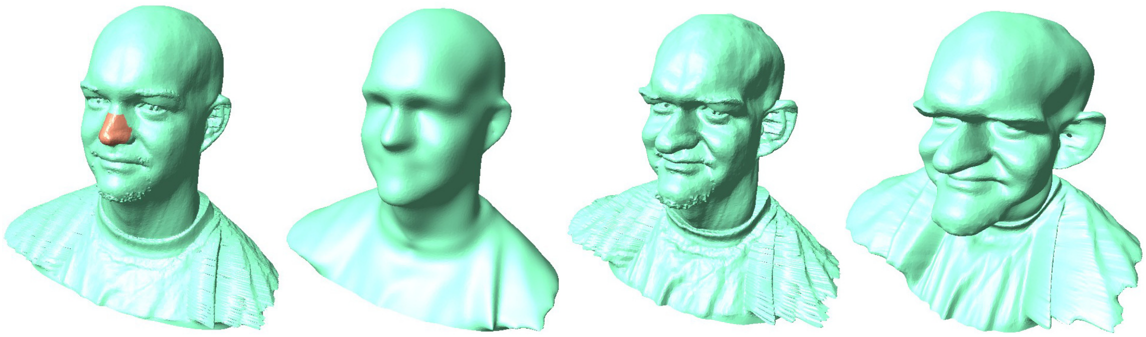

Figure 12: *Various filtering results for a human model (75,948 vertices, 151,474 triangles). Far left is the original model. The next image shows a lowpass filtered model ($s_1 = 25, s_2 = 50, G_0 = 1, G_1 = 1.1, G_\infty = 0$, 366 sec.). The last two are exaggerated models that look older ($s_1 = 200, s_2 = 1000, G_0 = 0.5, G_1 = 1.1, G_\infty = 0$,70 sec.) or like a cartoon character ($s_1 = 25, s_2 = 400, G_0 = 0.5, G_1 = 1.8, G_\infty = 0$, 481 sec.). Since the last two models have $G_0 = 0.5$, the filtered models are about half the size of the initial model. In this figure, we zoomed them in.*

8. Conclusion

We propose a mesh filtering framework that has physically meaningful parameters. A discrete Laplacian operator that approximates operator for smooth surfaces allows the user to compute filter parameters from the physical property of the surface such as feature size. We also propose to combine the explicit and implicit filter forms. Using this generalization, we have built a new filter that can be used for lowpass, highpass, bandpass, notch filters with various exaggeration and attenuation options that create a variety of effects on the mesh.

References

[Bus92] BUSHER P.: *Geometry and Spectra of Compact Riemann Surfaces*. Birkhauser Boston, 1992. 3

[Cun92] CUNNINGHAM E. P.: *Digital Filtering: An Introduction*. Houghton Mifflin, 1992. 4

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of ACM SIGGRAPH* (1999), pp. 317–324. 1, 2, 3, 4

[GLM96] GOTTSCHALK S., LIN M. C., MANOCHA D.: Obbtree: A hierarchical structure for rapid interference detection. In *Proceedings of ACM Siggraph* (1996). 7

[GSS99] GUSKOV I., SWELDENS W., SCHRÖDER P.: Multiresolution signal processing for meshes. In *SIGGRAPH* (1999). 2, 4

[KCVS98] KOBBELT L., CAMPAGNA S., VORSATZ J., SEIDEL H.-P.: Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of ACM SIGGRAPH* (1998), pp. 105–114. 4

[KG00] KARNI Z., GOTSMAN C.: Spectral compression of mesh geometry. In *Proceedings of ACM SIGGRAPH* (2000), pp. 279–286. 4

[NGH04] NI X., GARLAND M., HART J. C.: Fair morse functions for extracting the topological structure of a surface mesh. In *Proceedings of ACM SIGGRAPH* (2004), pp. 613–622. 5

[OBB00] OHTAKE Y., BELYAEV A. G., BOGAEVSKI I. A.: Polyhedral surface smoothing with simultaneous mesh regularization. In *Proceedings of the Geometric Modeling and Processing* (2000), pp. 229–237. 2, 4

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1 (1993), 15–36. 2

[Shi78] SHINNERS S. M.: *Modern Control System Theory and Application*. Addison Wesley, 1978. 6

[SK01] SCHNEIDER R., KOBBELT L.: Geometric fairing of irregular meshes for free-form surface design. *Computer Aided Geometric Design 18*, 4 (2001), 359–379. 2

[Tau95] TAUBIN G.: Signal processing approach to fair surface design. In *Proceedings of ACM SIGGRAPH* (1995), pp. 351–358. 1, 2, 3, 4

[TZG96] TAUBIN G., ZHANG T., GOLUB G.: Optimal surface smoothing as filter design. In *Fourth European Conference on Computer Vision (ECCV'96) and IBM Research Technical Report RC-20404* (March 1996). 1

[Xu04] XU G.: The convergent discrete laplace-beltrami operator over triangular surfaces. In *Proceedings of Geometric Modelling and Processing (GMP2004)* (2004), pp. 195–204. 2

[ZF03] ZHANG H., FIUME E.: Butterworth filtering and implicit fairing of irregular meshes. In *Proceedings of Pacific Graphics* (2003), pp. 502–506. 1, 2, 4, 5

[ZG04] ZELINKA S., GARLAND M.: Similarity-based surface modelling using geodesic fans. In *Proceedings of the 2nd Eurographics Symposium on Geometry Processing* (2004). 4