Spectral Geometry Processing with Manifold Harmonics

Bruno Vallet, Bruno Lévy

Spectral Geometry Processing with Manifold Harmonics

Bruno Vallet and Bruno Lévy
Tech Report ALICE-2007-001

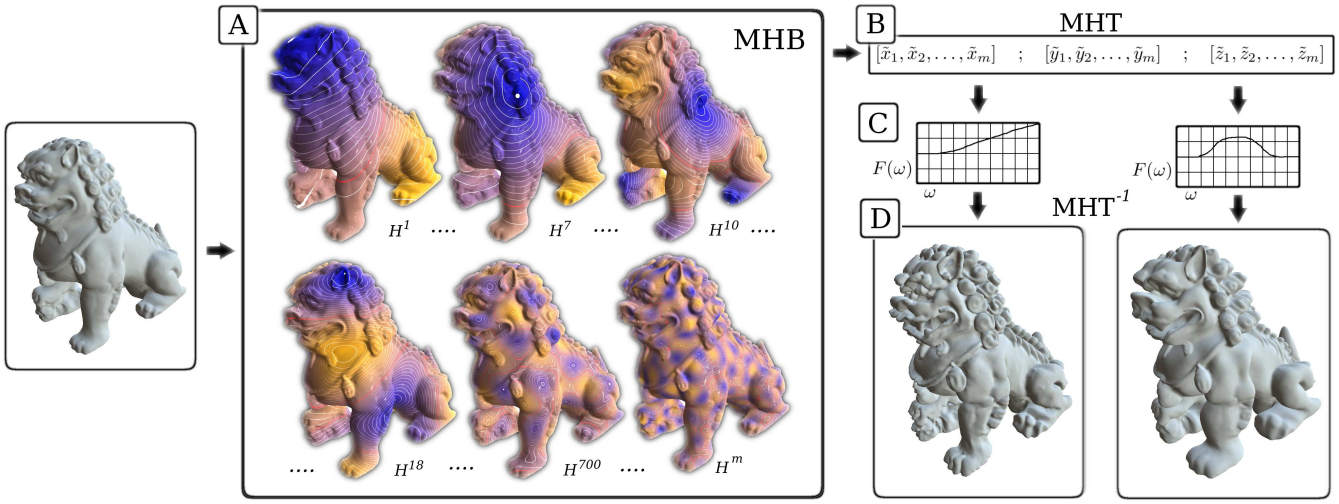

Figure 1: *Processing pipeline of our method: A: Compute the Manifold Harmonic Basis (MHB) of the input triangulated mesh. B: Transform the geometry into frequency space by computing the Manifold Harmonic Transform (MHT). C: Apply the frequency space filter on the transformed geometry. D: Transform back into geometric space by computing the inverse MHT.*

Abstract

We present a new method to convert the geometry of a mesh into frequency space. The eigenfunctions of the Laplace-Beltrami operator are used to define Fourier-like function basis and transform. Since this generalizes the classical Spherical Harmonics to arbitrary manifolds, the basis functions will be called *Manifold Harmonics*. It is well known that the eigenvectors of the discrete Laplacian define such a function basis. However, important theoretical and practical problems hinder us from using this idea directly. From the theoretical point of view, the combinatorial graph Laplacian does not take the geometry into account. The discrete Laplacian (*cotan weights*) does not have this limitation, but its eigenvectors are not orthogonal. From the practical point of view, computing even just a few eigenvectors is currently impossible for meshes with more than a few thousand vertices.

In this paper, we address both issues. On the theoretical side, we show how the FEM (Finite Element Modeling) formulation defines a function basis which is both geometry-aware and orthogonal. On the practical side, we propose a band-by-band spectrum computation algorithm and an out-of-core implementation that can compute thousands of eigenvectors for meshes with up to a million vertices. Finally, we demonstrate some applications of our method to interactive convolution geometry filtering and interactive shading design.

CR Categories: Computational Geometry and Object Modeling I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transformations

Keywords: Laplacian, Spectral Geometry, Filtering

Introduction

3D scanning technology easily produces computer representations from real objects. However, the acquired geometry often presents some noise that needs to be filtered out. More generally, it may be suitable to enhance some details while removing other ones, depending on their sizes, i.e. depending on the corresponding spatial frequencies. In his seminal paper, Taubin [1995] showed that the formalism of signal processing can be successfully applied to geometry processing. His approach is based on the similarity between the eigenvectors of the graph Laplacian and the basis functions used in the discrete Fourier transform. This Fourier function basis enables a given signal to be decomposed into a sum of sine waves of increasing frequencies. This analogy was used by Taubin as a theoretical tool to design and analyse an approximation of a low-pass filter. Several variants of this approach were then suggested, as discussed below.

In this paper, instead of using Fourier analysis as a theoretical tool to analyse approximations of filters, our idea is to fully generalize it to surfaces of arbitrary topology, and use it to achieve interactive general convolution filtering. Our processing pipeline is outlined in Figure 1:

- A: given a triangulated mesh with n vertices, compute a function basis $H^k, k = 1 \ldots m$ that we call the *Manifold Harmonics Basis* (MHB). The k^{th} element of the MHB is a piecewise linear function given by its values H^k_i at vertices i of the surface;
- B: once the MHB is computed, transform the geometry into frequency space by computing the Manifold Harmonic Transform (MHT) of the geometry, that is to say three vectors of coefficients $[\tilde{x}_1, \tilde{x}_2, \ldots \tilde{x}_m]$, $[\tilde{y}_1, \tilde{y}_2, \ldots \tilde{y}_m]$, and $[\tilde{z}_1, \tilde{z}_2, \ldots \tilde{z}_m]$.
- C: apply a frequency space filter $F(\omega)$ by multiplying each $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$ by $F(\omega_k)$, where ω_k denotes the frequency associated with H^k;
- D: finally, transform the object back into geometric space by applying the inverse MHT.

Note that this pipeline is similar to signal processing with the discrete cosine transform or with spherical harmonics. The main difference is that in our case, the MHB function basis, specially adapted to the surface, is not known in closed form. Therefore it requires additional computations to be constructed (step A). However, note that the MHB is directly attached to the input mesh, thus no resampling is needed. Once the MHB is known, the subsequent stages of the pipeline can be very efficiently computed. This allows the solution to be interactively updated when the $F(\omega)$ filter is modified by the user.

Contributions

- ◇ based on Finite Element Modeling, we give the complete derivation of a discrete Laplace-Beltrami operator which is both geometry-aware and orthogonal, and that generalizes the classical cotan weights;
- ◇ to compute the eigenfunctions, we propose a numerical solution mechanism that overcomes the current limits (thousands vertices) by several orders of magnitude (up to a million vertices). This makes spectral analysis directly usable in practice, besides its common use as a theoretical tool;
- ◇ our method can interactively filter functions defined over meshes. We demonstrate it applied to geometry and shading.

Previous Work

Approximated low-pass filters Spectral analysis was first used as a theoretical tool to characterize the classical approximations of low-pass filters [Taubin 1995]. The stability of Taubin's method was later improved by using an implicit formulation in [Desbrun et al. 1999], that replaces the uniform discrete Laplacian with a more elaborate one, similar to Pinkall and Polthier's discrete Laplacian [1993] (cotan weights). More recently, an extension was proposed in [Kim and Rossignac 2005], that can compute a wider class of filters (e.g. band-exaggeration), by combining the explicit and implicit schemes to reach the different frequency bands involved in the filter. Our method can use an arbitrary user-defined filter, and offers in addition the possibility of interactively changing the filter (at the expense of storing the MHB).

Energy minimization Spectral analysis can also be used to characterize the approaches based on an energy minimization (e.g. [Mallet 1992]). These methods are called *discrete fairing* in [Kobbelt 1997; Kobbelt et al. 1998], in reference to their continuous-setting counterparts [Bloor and Wilson 1990]. Recently, a method was proposed [Nealen et al. 2006] to optimize both inner fairness (triangle shapes) and outer fairness (surface smoothness), by using a combination of the combinatorial Laplacian and the discrete Laplace-Beltrami operator. Both variations of the Laplace operators are discussed below.

Geometry Filtering This is the most natural application of spectral analysis to geometry processing. To implement this idea, several methods consist in putting the input surface in one-to-one correspondence with a simpler domain [Zhou et al. 2004], or to partition it into a set of simpler domains [Lee et al. 1998; Pauly and Gross 2001] on which it is easier to define a frequency space. Note that these methods generally need to resample the geometry, with the exception of Mousa *et al.* [2006] who directly compute the Spherical Harmonic Transform of a star-shaped mesh. It is also possible to extract the frequencies from a progressive mesh [Lee et al. 1998] and avoid resampling the geometry by using irregular subdivision [Guskov et al. 1999]. Our method directly computes the frequency-space for a surface of arbitrary topology, without needing any resampling nor segmentation.

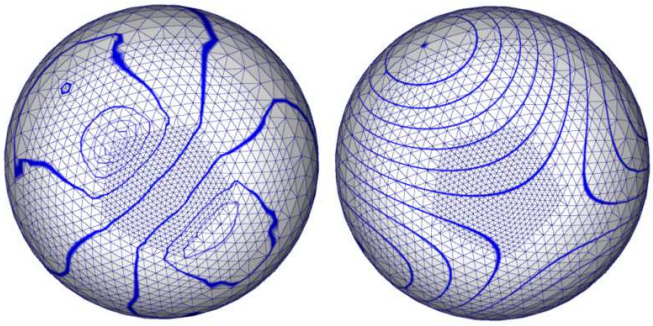

Figure 2: *Comparison of the iso-contours of the fourth eigenfunction on an irregularly sampled sphere. Left: combinatorial Laplacian. Right: geometry-aware discrete Laplacian (cotan weights).*

Combinatorial graph Laplacians It is well known that the eigenvectors of the combinatorial graph Laplacian define a Fourier-like function basis. Karni *et al.* [2000] proposed to use this idea for geometry compression. Since the eigenvectors of the combinatorial Laplacian are orthogonal, it is easy to project the geometry on them. To overcome the high computational cost associated with the eigenvectors computation, they partition the mesh into smaller charts and apply the method to each chart. Zhang [2004] studies several variants of the combinatorial graph Laplacian and their properties for spectral geometry processing and JPEG-like mesh compression. However, as pointed-out in [Meyer et al. 2003], the analogy between the graph Laplacian and the discrete cosine transform supposes a uniform sampling of the mesh. As a consequence, to make the eigenfunctions independent of the quality of the meshing, it is better to replace the combinatorial graph Laplacian with the discrete Laplacian operator.

Discrete Laplacian operator The combinatorial graph Laplacian solely depends on the connections between the vertices. As a consequence, two different embeddings of the same graph yield the same eigenfunctions (it does not take the geometry into account), and two different meshings of the same object yield different eigenfunctions (it is not independent of the mesh). Figure 2-Left shows the problem on an irregularly sampled sphere. The discrete Laplacian operator, i.e. the celebrated cotan weights [Pinkall and Polthier 1993; Meyer et al. 2003] does not suffer from this limitation (Figure 2-Right). It was recently used [Dong et al. 2006] to compute an eigenfunction and use it to steer a quad-remeshing process. In our setting, to separate the different frequencies of the shape we need to compute multiple eigenfunctions. In this context, the discrete Laplacian operator seemingly loses an important property of its continuous counterparts: since its coefficients are non-symmetric[1], the eigenvectors are no longer orthogonal. This makes the transform in frequency space difficult to compute (dense matrix invert instead of projection). A solution is to "symmetrize" the matrix [Levy 2006] at the expense of partially losing mesh independence. More general theoretical foundations are used by Reuter *et al.* [2005], who use FEM (Finite Element Modeling) to compute the spectrum (i.e. the eigenvalues), and use it as a signature for shape classification. Other works [Kim and Rossignac 2005; Kim and Rossignac 2006] also mention the possibility of using FEM. In this paper, we present a complete, standalone and simpler derivation of the FEM formulation for the eigenfunctions. We show how this generalizes the cotan weights in a way that preserves the two properties required by our spectral analysis (mesh independence and orthogonality).

[1]The denominator of coefficient $a_{i,j}$ is the area of vertex i's neighborhood, which may differ from the area of vertex j's neighborhood

The rest of the paper is organized as follows. We will first recall some notions on the Fourier Analysis and show how the Laplace operator and its eigenfunctions allow to generalize it to a spectral analysis on manifold (Section 1). A Manifold Harmonics Basis (MHB) will be built through a Finite Elements Discretization, and its relations with the classical discrete Laplacian will be explained in (Section 2). Equipped with this new tool, it is then simple to define the Manifold Harmonics Transform (MHT) that transforms from geometric space into frequency space, and the inverse MHT (Section 3). We will then explain how to compute the MHB efficiently in practice, and implement scalable spectral geometry processing (Section 4). We conclude by presenting some applications and results.

Before entering the heart of the matter, we introduce the Laplace operator, its generalizations, and its links with spectral analysis.

1 Spectral Analysis on Manifolds

Manifold harmonics (also called *shape harmonics*) are defined as the eigenfunctions of the Laplace operator. This section starts by defining the Laplace operator and gives some intuition on its meaning and importance. We first recall the more familiar Fourier analysis, and then show how the eigenfunctions of the Laplace-Beltrami operator generalize this setting to arbitrary manifolds. Then, section 2 will explain how to compute them.

1.1 Fourier Analysis

As in Taubin's article [1995], we start by studying the case of a closed curve, but staying in the continuous setting. Given a square-integrable periodic function $f : x \in [0,1] \mapsto f(x)$, or a function f defined on a closed curve parameterized by normalized arclength, it is well known that f can be expanded into an infinite series of sines and cosines of increasing frequencies:

$$f(x) = \sum_{k=0}^{\infty} \tilde{f}_k H^k(x) \quad ; \quad \begin{cases} H^0 & = & 1 \\ H^{2k+1} & = & \cos(2k\pi x) \\ H^{2k+2} & = & \sin(2k\pi x) \end{cases} \qquad (1)$$

where the coefficients $\tilde{f}_k$ of the decomposition are given by:

$$\tilde{f}_k = < f, H^k > = \int_0^1 f(x) H^k(x) dx \qquad (2)$$

and where $< .,. >$ denotes the inner product (i.e. the "dot product" for functions defined on $[0,1]$). See [Arvo 1995] or [Levy 2006] for an introduction to functional analysis. The "Circle harmonics" basis H^k is orthonormal with respect to $< .,. >$: $< H^k, H^k > = 1$, $< H^k, H^l > = 0$ if $k \neq l$.

The set of coefficients $\tilde{f}_k$ (Equation 2) is called the Fourier Transform (FT) of the function f. Given the coefficients $\tilde{f}_k$, the function f can be reconstructed by applying the inverse Fourier Transform FT^{-1} (Equation 1). Our goal is now to generalize these notions to arbitrary manifolds. To do so, we can consider the functions H^k of the Fourier basis as the eigenfunctions of $-\partial^2/\partial x^2$: the eigenfunctions H^{2k+1} (resp. H^{2k+2}) are associated with the eigenvalues $(2k\pi)^2$:

$$-\frac{\partial^2 H^{2k+1}(x)}{\partial x^2} = (2k\pi)^2 \cos(2k\pi x) = (2k\pi)^2 H^{2k+1}(x)$$

This construction can be extended to arbitrary manifolds by considering the generalization of the second derivative to arbitrary manifolds, i.e. the Laplace operator and its variants, introduced below.

1.2 The Laplace operator and its generalizations

The Laplace operator (or Laplacian) plays a fundamental role in physics and mathematics. In $\mathbb{R}^n$, it is defined as the divergence of the gradient:

$$\Delta = \text{div grad} = \nabla.\nabla = \sum_i \frac{\partial^2}{\partial x_i^2}$$

Intuitively, the Laplacian generalizes the second order derivative to higher dimensions, and is a characteristic of the irregularity of a function as $\Delta f(P)$ measures the difference between $f(P)$ and its average in a small neighborhood of P.

Generalizing the Laplacian to curved surfaces require complex calculations. These calculations can be simplified by a mathematical tool named exterior calculus (EC) [2]. EC is a coordinate free geometric calculus where functions are considered as abstract mathematical objects on which operators act. To use these functions, we cannot avoid instantiating them in some coordinate frames. However, most calculations are simplified thanks to higher-level considerations. For instance, the divergence and gradient are known to be coordinate free operators, but are usually defined through coordinates. EC generalizes the gradient by d and divergence by δ, which are built independently of any coordinate frame (see Appendix A).

Using EC, the definition of the Laplacian can be generalized to functions defined over a manifold $\mathscr{S}$ with metric g, and is then called the Laplace-Beltrami operator:

$$\Delta = \text{div grad} = \delta d = \sum_i \frac{1}{\sqrt{|g|}} \frac{\partial}{\partial x_i} \sqrt{|g|} \frac{\partial}{\partial x_i}$$

where $|g|$ denotes the determinant of g. The additional term $\sqrt{|g|}$ can be interpreted as a local "scale" factor since the local area element dA on $\mathscr{S}$ is given by $dA = \sqrt{|g|} dx_1 \wedge dx_2$.

Finally, for the sake of completeness, we can mention that the Laplacian can be extended to k-forms and is then called the Laplace-de Rham operator defined by $\Delta = \delta d + d\delta$. Note that for functions (i.e. 0-forms), the second term $d\delta$ vanishes and the first term δd corresponds to the previous definition.

We will now define the eigenfunctions of the Laplacian, and explain how they allow to generalize important concepts to arbitrary manifolds and triangulated meshes.

1.3 Laplacian Eigenfunctions

The eigenfunctions and eigenvalues of the Laplacian on a (manifold) surface $\mathscr{S}$, are all the pairs (H^k, λ_k) that satisfy:

$$-\Delta H^k = \lambda_k H^k \qquad (3)$$

The "$-$" sign is here required for the eigenvalues to be positive. On a closed curve, the eigenfunctions of the Laplace operator define the function basis (sines and cosines) of Fourier analysis, as recalled in Section 1.1. On a square, they correspond to the function basis of the DCT (Discrete Cosine Transform), used for instance by the JPEG image format. Finally, the eigenfunctions of the Laplace-Beltrami operator on a sphere define the Spherical Harmonics basis.

[2] To our knowledge, besides Hodge duality used to compute minimal surfaces [Pinkall and Polthier 1993], one of the first uses of EC in geometry processing [Gu and Yau 2002] applied some of the fundamental notions involved in the proof of Poincaré's conjecture to global conformal parameterization. More recently, a Siggraph course was given by Schroeder *et al.* [2005], making these notions usable by a wider community.

Figure 3: *Some functions of the Manifold Harmonic Basis (MHB) on the Gargoyle dataset*

In these three simple cases, two reasons make the eigenfunctions a function basis suitable for spectral analysis of manifolds:

1. Because the Laplacian is symmetric ($<\Delta f, g>=<f, \Delta g>$), its eigenfunctions are orthogonal, so it is extremely simple to project a function onto this basis, i.e. to apply a Fourier-like transform to the function.

2. For physicists, the eigenproblem (Equation 3) is called the Helmoltz equation, and its solutions H^k are stationary waves. This means that the H^k are functions of constant wavelength (or spatial frequency) $\omega_k = \sqrt{\lambda_k}$.

Hence, using the eigenfunctions of the Laplacian to construct a function basis on a manifold is a natural way to extend the usual spectral analysis to this manifold. In our case, the manifold is a mesh, so we need to port this construction to the discrete setting. The first idea that may come to the mind is to apply spectral analysis to a discrete Laplacian matrix (e.g. the cotan weights). However, the discrete Laplacian is not a symmetric matrix (the denominator of the $a_{i,j}$ coefficient is the area of vertex $i's$ neighborhood, that does not necessarily correspond to the area of vertex j's neighborhood). Therefore, we lose the symmetry of the Laplacian and the orthogonality of its eigenvectors. This makes it difficult to project functions onto the basis. For this reason, we will clarify the relations between the continuous setting (with functions and operators), and the discrete one (with vectors and matrices) in the next section.

2 The Manifold Harmonic Basis

To be able to project onto the eigenfunctions, we need to solve our eigenproblem (Equation 3) numerically in a way that preserves their orthogonality. Since it is based on the theory of Hilbert spaces, structured by the inner product $<.,.>$, Finite Element Modeling (FEM) gives a solid theoretical foundation that meets this requirement. We call *Manifold Harmonics Basis* (MHB) the solutions of the FEM formulation. We show how this relates with the classical discrete Laplacian and how orthogonality can be recovered.

2.1 Finite element formulation

To setup our finite element formulation, we first need to define a set of *basis functions* used to express the solutions, and a set of *test functions* onto which the eigenproblem (Equation 3) will be projected. As it is often done in FEM, we choose for both basis and test functions the same set $\Phi^i(i = 1 \dots n)$. We use the "hat" functions (also called P_1), that are piecewise-linear on the triangles, and that are such that $\Phi^i(i) = 1$ and $\Phi^i(j) = 0$ if $i \neq j$. Geometrically, Φ^i corresponds to the barycentric coordinate associated with vertex i on each triangle containing i. Solving the finite element formulation of Equation 3 relative to the Φ^i's means looking for functions of the form: $H^k = \sum_{i=1}^{n} H_i^k \Phi^i$ which satisfy Equation 3 in projection on the Φ^j's:

$$\forall j, <-\Delta H^k, \Phi^j> = \lambda_k <H^k, \Phi^j>$$

or in matrix form:

$$-Q\mathbf{h}^k = \lambda B\mathbf{h}^k \tag{4}$$

where $Q_{i,j} =<\Delta\Phi^i, \Phi^j>$, $B_{i,j} =<\Phi^i, \Phi^j>$ and where $\mathbf{h}^k$ denotes the vector $[H_1^k, \dots H_n^k]$. The matrix Q is called the *stiffness matrix*, and B the *mass matrix*. The detailed derivations are provided in Appendix B and lead to:

$$\begin{cases} Q_{i,j} &= \left(\text{cotan}(\beta_{i,j}) + \text{cotan}(\beta'_{i,j})\right)/2 \\ Q_{i,i} &= -\sum_j Q_{i,j} \end{cases}$$
$$\begin{cases} B_{i,j} &= (|t| + |t'|)/12 \\ B_{i,i} &= (\sum_{t \in St(i)} |t|)/6 \end{cases} \tag{5}$$

where t, t' are the two triangles that share the edge (i, j), $|t|$ and $|t'|$ denote their areas, $\beta_{i,j}$, $\beta'_{i,j}$ denote the two angles opposite to the edge (i, j) in t and t', and $St(i)$ denotes the set of triangles incident to i (see also Figure 12 in the Appendix).

To simplify the computations, a common practice of FEM consists in replacing this equation with an approximation:

$$-Q\mathbf{h}^k = \lambda D\mathbf{h}^k \quad \left(\text{or} \quad -D^{-1}Q\mathbf{h}^k = \lambda\mathbf{h}^k\right) \tag{6}$$

where the mass matrix B is replaced with a diagonal matrix D called the *lumped mass matrix*, and defined by:

$$D_{i,i} = \sum_j B_{i,j} = (\sum_{t \in St(i)} |t|)/3. \tag{7}$$

Note that D is non-degenerate (as long as mesh triangles have non-zero areas). FEM researchers [Prathap 1999] explain that besides simplifying the computations this approximation fortuitously improves the accuracy of the result, due to a cancellation of errors, as pointed out in [Dyer 2006]. The practical solution mechanism to solve Equation 6 will be explained further in Section 4, and Figure 3 shows some of its solutions on the Gargoyle dataset.

Remark: The matrix $D^{-1}Q$ in (Equation 6) exactly corresponds to the usual discrete Laplacian (cotan weights). Hence, in addition to direct derivation of triangle energies [Pinkall and Polthier 1993] or averaged differential values [Meyer et al. 2003], the discrete Laplacian can be derived from a lumped-mass FEM formulation. As will be seen in the next section, the FEM formulation and associated inner product will help us understand why the orthogonality of the eigenvectors seems to be lost (since $D^{-1}Q$ is not symmetric), and how to retrieve it.

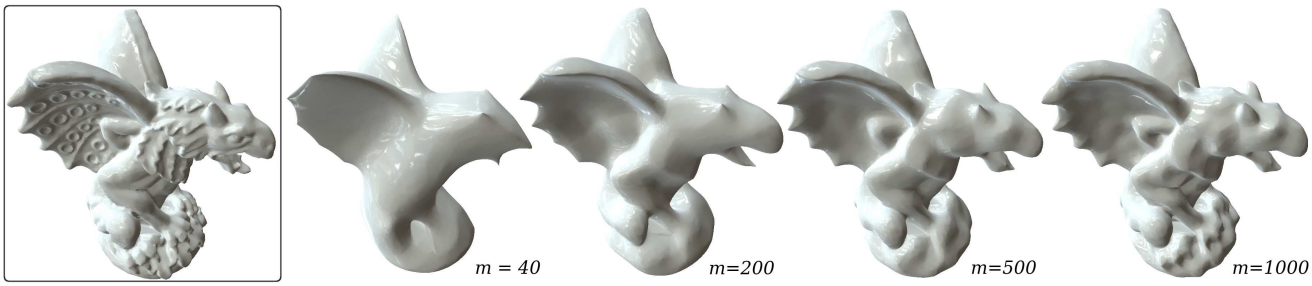

Figure 4: *Reconstructions obtained with an increasing number of MH functions.*

2.2 Orthogonality of the MHB

The Manifold Harmonic Transform is based on a projection onto the MHB, for which we need the basis to be orthonormal. Thus we need to clarify how the orthogonality of the continuous Laplacian Δ is preserved by the discretization. In fact, we just have to make the distinction between the (discrete) vector dot product $\mathbf{g}^\top\mathbf{h}$ and the (continuous) function inner product $< G,H >$:

$$< G,H >=< \sum_{i=1}^{n} G_i\Phi^i, \sum_{j=1}^{n} H_j\Phi^j >$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} G_iH_j < \Phi^i,\Phi^j >= \sum_{i=1}^{n}\sum_{j=1}^{n} G_iH_jB_{i,j} = \mathbf{g}^\top B\mathbf{h}$$

The vector dot product and function inner product coincide only if $B = Id$, that is if the basis of test functions (Φ^i) is orthonormal. This is not true in our case (the "hat functions" are not orthogonal), therefore we need to use the inner product $\mathbf{g}^\top B\mathbf{h}$.

With the lumped-mass approximation, the inner product is given by $\mathbf{g}^\top D\mathbf{h}$, and two eigenvectors $\mathbf{g}$ and $\mathbf{h}$ associated with different eigenvalues satisfy $\mathbf{g}^\top D\mathbf{h} = 0$. In addition to $D-$orthogonality, it is easy to ensure that the MHB is orthonormal, by dividing each vector $\mathbf{h}^k$ by its D-relative norm $\|\mathbf{h}^k\|_D = (\mathbf{h}^{k\top} D\mathbf{h}^k)^{1/2}$.

To summarize, solving for the eigenfunctions of the Laplacian using the Finite Element Approximation and the lumped-mass approximation reduces to the matricial eigenproblem (Equation 6). The practical solution mechanism for this eigenproblem is provided in Section 4 and yields a series of eigenpairs $(\mathbf{h}^k = [H_1^k, H_2^k, \ldots H_n^k], \lambda_k)$ called the Manifold Harmonics Basis (MHB). The MHB along with the adequate inner product will now allow us to define a Manifold Harmonic Transform (MHT) and inverse MHT.

3 The Manifold Harmonic Transform

Now that we have computed the MHB by solving equation 6, and that we have understood the difference between dot and inner products, we can give the expressions of the MHT (from geometric space to frequency space) and inverse MHT (from frequency space to geometric space). We will also explain how they can be used to implement geometric filtering.

3.1 Computing the MHT

The geometry x (resp. y,z) of the triangulated surface $\mathscr{S}$ can be seen as a piecewise linear function defined as a linear combination of the basis functions Φ^i: $x = \sum_{i=1}^{n} x_i\Phi^i$ where x_i denotes the x coordinate at vertex i.

Computing the MHT of the function x means converting x from the "hat functions" (Φ^i) basis (geometric space) into the MHB (H^k) (frequency space). Since the MHB is orthonormal, this can be done by projecting x onto the MHB through the inner product. The MHT of x is a vector $[\tilde{x}_1, \tilde{x}_2, \ldots \tilde{x}_m]$, given by:

$$\tilde{x}_k =< x, H^k >= \mathbf{x}^\top D\mathbf{h}^k = \sum_{i=1}^{n} x_iD_{i,i}H_i^k \tag{8}$$

where $\mathbf{x}$ denotes the vector $x_1, x_2, \ldots x_n$.

3.2 Computing the inverse MHT

The inverse MHT, that maps from frequency space into geometric space, is given by the expression of x in the MHB (H^k). The reconstructed coordinate x at a vertex i is then given by :

$$x_i = \sum_{k=1}^{m} \tilde{x}_kH_i^k \tag{9}$$

Figure 4 shows the geometry reconstructed from the MHT of a surface using a different number m of MHT coefficients. As Figure 4 shows, the first H^k functions capture the general shape of the functions without any shrinking effect and the next ones correspond to the details. Geometric filtering can then be implemented by modifying the inverse MHT.

3.3 Filtering

Once the geometry is converted in the MHB, each component $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$ of the MHT correspond to an individual spatial frequency ω_k. In the case of a closed curve (Section 1.1), we have $-\partial^2 \sin(\omega x)/\partial x^2 = \omega^2 \sin(\omega x)$, therefore the relation between the spatial frequency ω_k and the associated eigenvalue λ_k is $\omega_k = \sqrt{\lambda_k}$. This still holds for the eigenfunctions of the Laplace operator on a surface (Section 1.3).

A frequency-space filter is a function $F(\omega)$ that gives the amplification to apply to each spatial frequency ω. Since all frequencies are separated by the MHT, applying a filter $F(\omega)$ to the geometry becomes a simple product in frequency space, such that the filtered coordinate x_i^F (resp y_i^F, z_i^F) of vertex i is given by:

$$x_i^F = \sum_{k=1}^{m} F(\omega_k)\tilde{x}_kH_i^k = \sum_{k=1}^{m} F(\sqrt{\lambda_k})\tilde{x}_kH_i^k$$

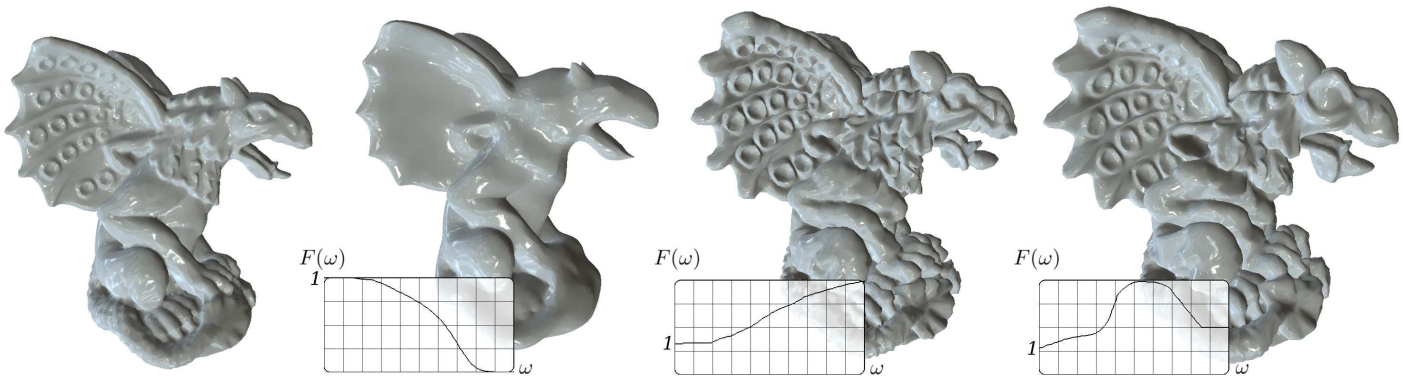

Figure 5: *Low-pass, enhancement and band-exaggeration filters. The filter can be changed by the user, the surface is updated interactively.*

In practice, since the MHB stops at frequency $\omega_m = \sqrt{\lambda_m}$, smaller geometric details are not represented in the MHT. However, it is possible to keep track of all the high-frequency information, by storing in each vertex the difference x_i^{hf} (resp. y_i^{hf}, z_i^{hf}) between the original geometry and the projection onto the MHB given by:

$$x_i^{hf} = \sum_{k=m+1}^{\infty} \tilde{x}_k H_i^k = x_i - \sum_{k=1}^{m} \tilde{x}_k H_i^k$$

The frequency space filter can be applied to the high-frequency components of the signal, by re-injecting them into the inverse MHT, as follows:

$$x_i^F = \sum_{k=1}^{m} F(\omega_k)\tilde{x}_k H_i^k + f^{hf} x^{hf} \text{ where } f^{hf} = \frac{1}{\omega_M - \omega_m} \int_{\omega_m}^{\omega_M} F(\omega) d\omega \tag{10}$$

In this equation, the term f^{hf} denotes the average value of the filter F on $[\omega_m, \omega_M]$, where ω_M denotes the maximum (Nyquist) frequency of the mesh (twice the edge length). The high-frequency component behaves like a wave packet that can be filtered as a whole, but that cannot be considered as independent frequencies. In our experiments, we used 10 times the average edge length to define the cutoff frequency ω_m.

Figure 5 demonstrates low-pass, enhancement and band-exaggeration filters. Note that arbitrary frequencies can be filtered without any shrinking effect. Moreover, only the filtered inverse MHT (Equation 10) depends on the filter F. As a consequence, by storing the MHB and the MHT, the solution can be updated interactively when the user changes the filter F.

We now proceed to explain how to compute the coefficients H_i^k of the MHB and the associated eigenvalues λ_k.

4 Numerical Solution Mechanism

Computing the MHB means solving for the eigenvalues λ_k and eigenvectors $\mathbf{h}^k$ for the matrix $-D^{-1}Q$:

$$-D^{-1}Q\mathbf{h}^k = \lambda_k \mathbf{h}^k$$

However, eigenvalues and eigenvectors computations are known to be extremely computationally intensive. To reduce the computation time, Karni *et al.* [2000] partition the mesh into smaller charts, and [Dong et al. 2006] use multiresolution techniques. In our case, we need to compute multiple eigenvectors (typically a few thousands). This is known to be currently impossible for meshes with more than a few thousand vertices [Wu and Kobbelt 2005]. In this section, we show how this limit can be overcome by several order of magnitudes.

To compute the solutions of a large sparse eigenproblems, several iterative algorithms exist. The publically available library ARPACK (used in [Dong et al. 2006]) provides an efficient implementation of the Arnoldi method. Yet, two characteristics of eigenproblem solvers hinder us from using them directly to compute the MHB for surfaces with more than a few thousand vertices:

- ◇ first of all, we are interested in the lower frequencies, i.e. eigenvectors with associated eigenvalues lower than ω_m^2. Unfortunately, the iterative solver performs much better for the other end of the spectrum. This contradicts intuition as in mechanical simulations for instance, it is difficult to ensure the stability of numerical schemes in high frequencies. However, this can be explained in terms of filtering as lower frequencies correspond to higher powers of the smoothing kernel, which may have a poor condition number;
- ◇ secondly, we need to compute a large number of eigenvectors (typically a thousand), and it is well known that computation time is superlinear in the number of requested eigenpairs. In addition, if the surface is large (millions vertices), the MHB does not fit in system RAM.

We address both issues by applying spectral transforms to the eigenproblem. To get the eigenvectors of a spectral band centered around a value λ_S, we start by shifting the spectrum by λ_S, by replacing $-D^{-1}Q$ with $-D^{-1}Q - \lambda_S Id = -D^{-1}(Q + \lambda_S D)$. Then, we can swap the spectrum by inverting this matrix. This is called the *shift-invert* spectral transform, and the new eigenproblem to solve is given by:

$$-(Q + \lambda_S D)^{-1} D\mathbf{h}^k = \mu_k \mathbf{h}^k$$

It is easy to check that its eigenvectors are the same as the original ones, and that the eigenvalues are given by $\lambda_k = \lambda_S + 1/\mu_k$. This gives a band centered around λ_S as iterative solvers return the high end of the spectrum (largest μ's). It is then possible to split the MHB computation into multiple bands, and obtain a computation time that is linear in the number of computed eigenpairs. In addition, if the MHB does not fit in RAM, each band can be streamed into a file.

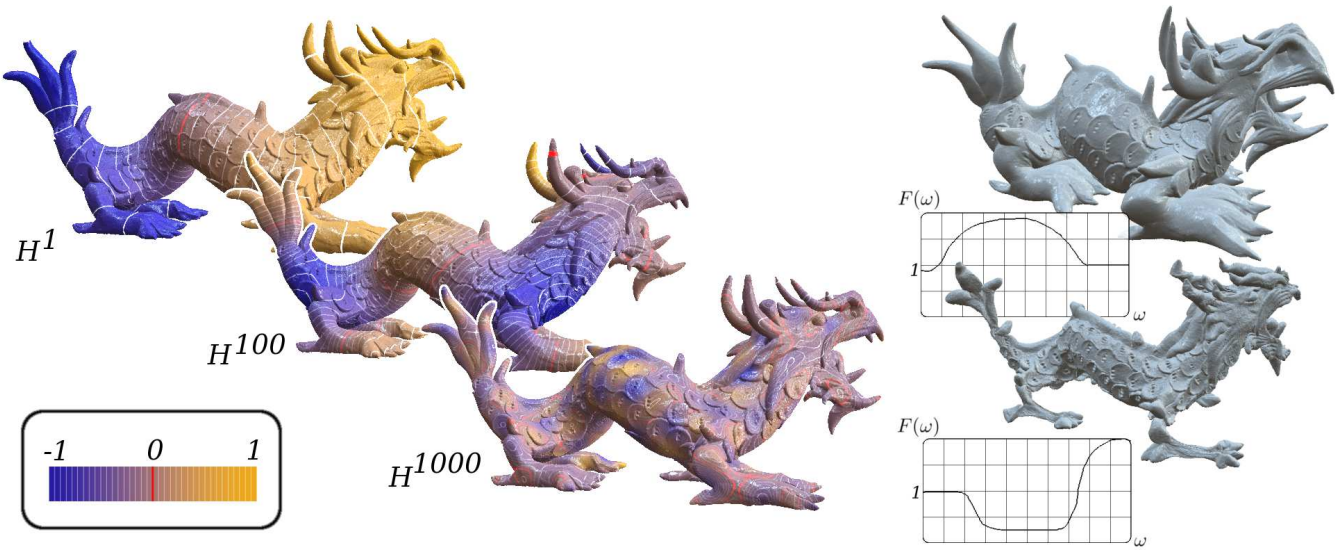

Figure 6: *Toward scalable spectral geometry processing: the MHB computed on 1M vertices (XYZ dragon) and OOC convolution filtering.*

The band-by-band algorithm can then be detailed:

```
(1)   λ_S ← 0   ;   λ_last ← 0
(2)   while(λ_last < ω_m^2)
(3)     compute an inverse M of (Q + λ_S D)
(4)     find the 50 first eigenpairs (h^k, μ_k) of −MD
(5)     for k = 1 to 50
(6)       λ_k ← λ_S + 1/μ_k
(7)       if (λ_k > λ_last) write(h^k, λ_k)
(8)     end //for
(9)     λ_S ← λ_50 + 0.4(λ_50 − λ_1)
(10)    λ_last ← λ_50
(11)  end //while
```

Before calling the eigen solver, we pre-compute the inverse M of $Q+\lambda_S D$ with a sparse direct solver (Line 3). The fact that $Q+\lambda_S D$ may be singular (for instance, if $\lambda_S = 0$, the vector $[1,1,\ldots 1]$ is in its kernel) is not a problem since the spectral transform is still valid when using an *indefinite* factorization. The factorized $Q+\lambda_S$ is used in the inner loop of the eigen solver (Line 4). To factorize $Q+\lambda_S$, we used the sparse OOC (out-of-core) symmetric indefinite factorization [Meshar et al. 2006] implemented in the future release of TAUCS, kindly provided by S. Toledo. We then recover the λ's from the μ's (Line 6) and stream-write the new eigenpairs into a file (Line 7). Since the eigenvalues are centered around the shift λ_S, the shift for the next band is given by the last computed eigenvalue plus slightly less than half the bandwidth to ensure that the bands overlap and that we are not missing any eigenvalue (Line 9).

Note that ARPACK implements the shift-invert spectral transform. However, since we use a direct solver, it is more efficient to use our own implementation of the spectral transform as recommended in ARPACK's user guide.

We have experimented the OOC factorization combined with the streamed band-by-band eigenvectors algorithm for computing up to a thousand eigenvectors on a mesh with one million vertices. We have also implemented an OOC version of the MHT, filtering and inverse MHT, that reads one frequency band at a time and accumulates its contribution (Figure 6). For smaller meshes (hundreds thousands vertices), a faster in-core sparse factorization can be used. Note that before the next release of TAUCS is available, the reader who wants to reproduce our results may use SuperLU instead (at the expense of losing scalability).

	n	m	MHB	MHT	MHT^{-1}
gargoyle (Fig. 5)	25K	1340	175 s.	0.38 s.	0.51 s.
dino (Fig. 8)	56K	447	137 s.	0.34 s.	0.53 s.
dragon (Fig. 1)	150K	315	370 s.	0.65 s.	1.02 s.
XYZ dragon 1 (*)	244K	667	17 m. 12 s.	18 s.	4 s.
XYZ dragon 2 (**)	500K	800	4 h. 12 m.	32 s.	48 s.
XYZ dragon 3 (**)	1M	1331	10 h. 35 m.	76 s.	85 s.

Table 1: *Timings for the different phases of the algorithm. For each data set, we give the number of vertices n, the number of computed eigenfunctions m, and the timings for the MHB, MHT and inverse MHT with filtering (Intel T7600 2.33 GHz). The symbol (*) indicates that the OOC MHT is used, and (**) indicates that both OOC factorization and OOC MHT are used.*

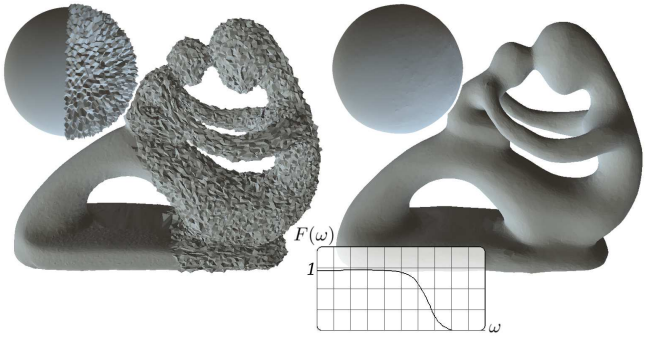

Figure 7: *Left: a sphere and a genus-4 model with random noise added. Right: the low-pass filtered result.*

Results and Conclusions

We have experimented our filtering method with object of different sizes. The timings are reported in table 1. Our MH-based filtering can be applied to objects of arbitrary topology. Figure 7 shows a low-pass filter used to remove high-frequency noise from a sphere and from a genus 4 object. The low-pass filter nearly preserves the symmetry of the sphere. Figure 8 and the video show how our method implements an interactive version of *geofilter* [Kim and Rossignac 2005]. In addition, since we are not using any approximation, our filter does not introduce any shrinking effect.

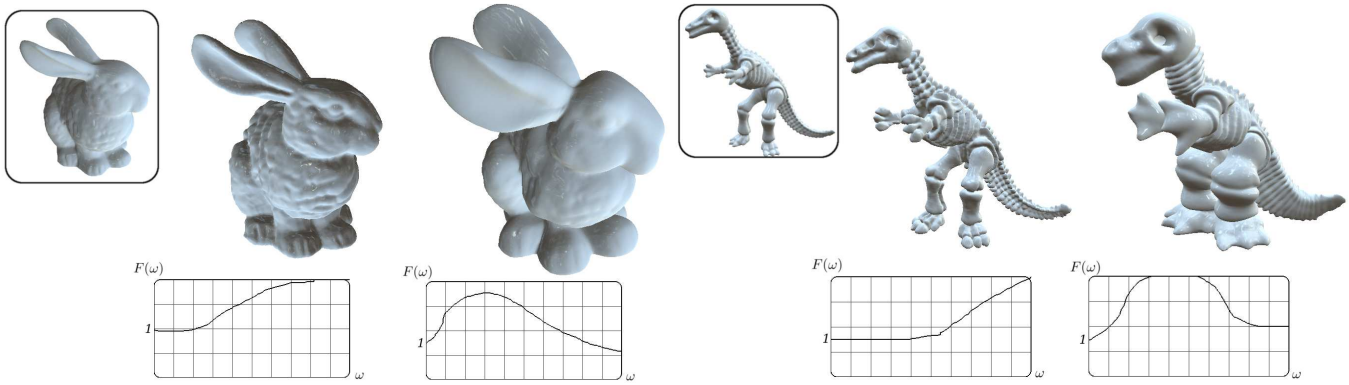

Figure 8: *Filtering Stanford's bunny and Cyberware's dinosaur. Results similar to geofilter are obtained, with the addition of interactivity.*

Figure 9: *Filtering the colors attached to the vertices of an object of arbitrary topology.*

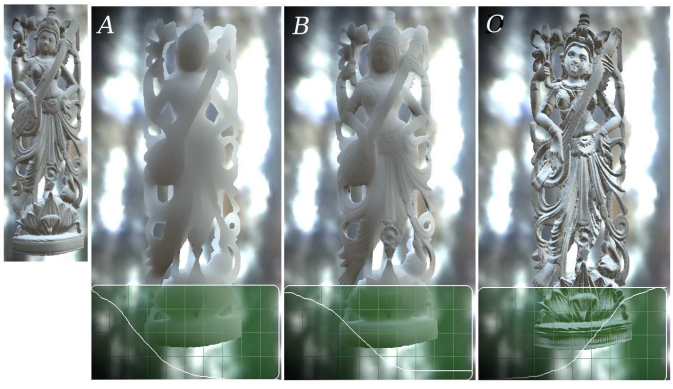

Figure 10: *Signal processing approach to shading design. A: high scattering; B: moderate scattering; C: exaggerated shading*

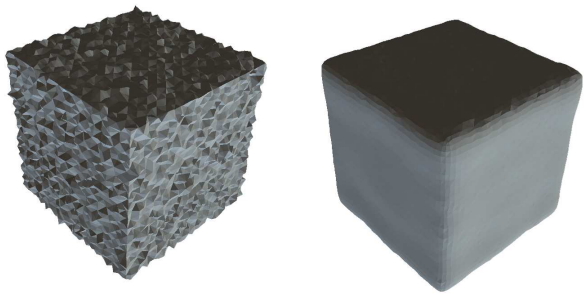

Figure 11: *Sharp creases yield harmonics of many different frequencies, and are therefore difficult to preserve when filtering.*

We demonstrate the versatility of our method, by applying it to various attributes attached to the vertices of surfaces. Figure 9 demonstrates our method applied to colors attached to the vertices of the mesh (enhancement and low-pass filters). Figure 10 shows how our framework applied to the normal vector can simulate various lighting effects. Applying a low-pass filter to the normal vector is approximatively equivalent to filtering light intensity. This yields a very simple approximation of subsurface scattering, that the user can easily tune by adjusting the filter (as shown in the video). Once the user is pleased with the result, only an additional normal vector is required to display the effect. The effect is simply obtained by replacing the normal vector with the filtered vector in the shader. Conversely, applying an enhancement filter to the normal vector yields a result very similar to the *exaggerated shading* method [Rusinkiewicz et al. 2006]. Note that the user can interactively generate any intermediate shading style between these two extremes.

Conclusions

In this paper, we have presented new methods for filtering functions defined on manifolds. We have given hindsight on the symmetry and orthogonality of discrete Laplace operators by separating the mass matrix from the stiffness matrix. We used our theoretical framework to define an orthonormal function basis localized in frequency space. On the practical side, we have overcome the current size limits of spectral geometry processing by several order of magnitudes, by making it usable for meshes with up to $10^5 - 10^6$ vertices. However, the main limitation of our method is that the storage space and pre-processing time for the MHB start to be expensive (hours) beyond 10^6 vertices. This will be optimized in future works, by introducing multiresolution in our solution mechanism.

Our implementation of MH-based geometry filtering computes the MHT of the x, y and z coordinates, that are dependent on the global orientation of the object. Therefore, we think that better results may be obtained by computing the MHT of some invariant local differential coordinates instead of using the absolute (x,y,z) coordinates.

Another limitation of our method concerns objects with creases. It is well known that low-pass filters based on Fourier-like methods cannot preserve the creases (Figure 11). Using the eigenfunctions of an anisotropic version of the Laplace operator may improve the frequency localization of the creases and therefore better preserve them when filtering.

Our method and associated numerical solution mechanism may find applications in various contexts, e.g. segmentation, mesh watermarking or reconstruction. Since our solver can process meshes with up to one million vertices, we have also experimented Karni *et al.*'s Spectral Mesh Compression [2000] without partitioning the object. It turned out that because the MHB is not spatially localized, many MHT coefficients (several thousands) were required to accurately reconstruct the geometry. Besides Karni *et al.*'s initial concern of reducing computation time, we think that partitioning also partially fixes the problem of spatial localization [3] at the expense of losing continuity. This leads to forecast that defining *Manifold Wavelets* localized in both frequency and spatial domains [Grinspun et al. 2002] will be an exciting research avenue in the future.

Acknowledgments

The acknowledgments will be given in the final version.

References

ARVO, J. 1995. The Role of Functional Analysis in Global Illumination. In *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, Springer-Verlag, New York, NY, P. M. Hanrahan and W. Purgathofer, Eds., 115–126.

BLOOR, M., AND WILSON, M. 1990. Using partial differential equations to generate free-form surfaces. *Computer-Aided Design*, 22, 202–212.

DESBRUN, M., MEYER, M., SCHRDER, P., AND BARR, A. H. 1999. Implicit fairing of arbitrary meshes using diffusion and curvature flow. In *SIGGRAPH Proceedings*.

DONG, S., BREMER, P.-T., GARLAND, M., PASCUCCI, V., AND HART, J. C. 2006. Spectral surface quadrangulation. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, ACM Press, New York, NY, USA, 1057–1066.

DYER, R. 2006. Mass weights and the cot operator. Tech. rep., Simon Fraser University, CA.

GRINSPUN, E., KRYSL, P., AND SCHRDER, P. 2002. Charms: a simple framework for adaptive simulation. In *Processings SIGGRAPH*.

GU, X., AND YAU, S.-T. 2002. Computing conformal structures of surfaces. *Communications in Information and Systems 2*, 2, 121–146.

GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. *Computer Graphics Proceedings (SIGGRAPH 99)*, 325–334.

KARNI, Z., AND GOTSMAN, C. 2000. Spectral compression of mesh geometry. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 279–286.

KIM, B., AND ROSSIGNAC, J. 2005. GeoFilter: Geometric Selection of Mesh Filter Parameters. *Computer Graphics Forum 24*, 3, 295–302.

KIM, B., AND ROSSIGNAC, J. 2006. Localized bi-Laplacian Solver on a Triangle Mesh and Its Applications. *Technical Report*.

[3] this is also why JPEG uses small blocks instead of applying the DCT to the whole image

KOBBELT, L., CAMPAGNA, S., VORSATZ, J., AND SEIDEL, H. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *SIGGRAPH Conference Proceedings*, 105–114.

KOBBELT, L. 1997. Discrete fairing. In *Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces*, 101–131.

LEE, A. W. F., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D. 1998. Maps: Multiresolution adaptive parameterization of surfaces. *Computer Graphics Proceedings (SIGGRAPH 98)*, 95–104.

LEVY, B. 2006. Laplace-beltrami eigenfunctions: Towards an algorithm that understands geometry. In *IEEE International Conference on Shape Modeling and Applications*.

MALLET, J. 1992. Discrete Smooth Interpolation. *Computer Aided Design 24*, 4, 263–270.

MESHAR, O., IRONY, D., AND TOLEDO, S. 2006. An out-of-core sparse symmetric indefinite factorization method. *ACM Transactions on Mathematical Software 32*, 445–471.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, Heidelberg, 35–57.

MOUSA, M., CHAINE, R., AND AKKOUCHE, S. 2006. Direct spherical harmonic transform of a triangulated mesh. *GRAPHICS-TOOLS 11*, 2, 17–26.

NEALEN, A., IGARASHI, T., SORKINE, O., AND ALEXA, M. 2006. Laplacian mesh optimization. In *Proceedings of ACM GRAPHITE*, 381–389.

PAULY, M., AND GROSS, M. 2001. Spectral processing of point sampled geometry. In *SIGGRAPH Proceedings*.

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1.

PRATHAP, G. 1999. Towards a science of fea: Patterns, predictability and proof through some case studies. *Current Science*.

REUTER, M., WOLTER, F.-E., AND PEINECKE, N. 2005. Laplace-spectra as fingerprints for shape matching. In *SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling*, ACM Press, New York, NY, USA, 101–106.

RUSINKIEWICZ, S., BURNS, M., AND DECARLO, D. 2006. Exaggerated shading for depicting shape and detail. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (July).

SCHROEDER, P., GRINSPUN, E., AND DESBRUN, M. 2005. Discrete differential geometry: an applied introduction. In *SIGGRAPH Course Notes*.

TAUBIN, G. 1995. A signal processing approach to fair surface design. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 351–358.

WU, J., AND KOBBELT, L. 2005. Efficient spectral watermarking of large meshes with orthogonal basis functions. In *The Visual Computer*.

ZHANG, H. 2004. Discrete combinatorial laplacian operators for digital geometry processing. In *Proc. SIAM Conference on Geometric Design and Computing*.

ZHOU, K., BAO, H., AND SHI, J. 2004. 3d surface filtering using spherical harmonics. In *Computer-Aided Design 36*, 363375.

A Exterior calculus on Manifolds

A.1 Chains and forms

We call $\wedge_k(\mathbb{R}^n)$ and $\wedge^k(\mathbb{R}^n)$ the spaces of k-chains and k-forms, which are defined as the skew symmetric $(0,k)$ and $(k,0)$ tensors (respectively). Intuitively, k-chains are oriented volume elements of dimension k: 0-chains are scalars, 1-chains are oriented lengths (vectors), 2-chains are oriented surface elements, 3-chains oriented volumes... k-forms are dual to k-chains in the sense that they are functions from k-chains to a field (usually $\mathbb{R}$). If we call $\partial_i = \partial/\partial x_i$ the canonic basis of $\mathbb{R}^n$, and ∂^i the dual basis, then a k-chain α_k and a k-form α^k write:

$$\alpha_k = \sum_I \alpha_k^I \partial_I \qquad \alpha^k = \sum_I \alpha_I^k \partial^I$$

where $I = \{i_1 ... i_k\}_{i_1 < ... < i_k}$ and $\partial_I = \partial_{i_1} \wedge ... \wedge \partial_{i_k}$, $\partial^I = \partial^{i_1} \wedge ... \wedge \partial^{i_k}$. ∂_I and ∂^I form basis of $\wedge_k(\mathbb{R}^n)$ and $\wedge^k(\mathbb{R}^n)$ which are vector spaces of dimension (n,k). Notations ∂_i and ∂^i come from differential geometry and are required to apply Einstein notation [4]. Moreover, it conveys the idea of duality between forms and chains.

A.2 Basic operators on forms

We recall here the definitions of the basic operators on k-forms. The expressions are exactly the same on k-chains by changing indices in exposants and vice versa. For each operator we will give the formal definition, and expression on coordinates using Einstein notation.

The wedge product $\wedge$ is defined as the only antisymmetric commutative linear operator from $\wedge_k(\mathbb{R}^n) \times \wedge_l(\mathbb{R}^n)$ to $\wedge_{k+l}(\mathbb{R}^n)$:

$$\alpha_k \wedge \alpha_l = \alpha_k^I \alpha_l^J \partial_I \wedge \partial_J$$

The exterior derivative $d : \wedge_k(\mathbb{R}^n) \to \wedge_{k+1}(\mathbb{R}^n)$ has an abstract definition through 4 properties which makes it unique: $d\alpha^0 = \partial_i \alpha^0 \partial^i$ (Einstein notation), $d(\alpha^k \wedge \beta^l) = d\alpha^k \wedge \beta^l + (-1)^k \alpha^k \wedge d\beta^l$, and $d \circ d = 0$. In coordinates, it writes:

$$d\alpha = \partial_i \alpha_I \partial^i \wedge \partial^I$$

From this definition, it can be proven that the exterior derivative satisfies:

$$\int_\Omega d\alpha = \int_{\partial\Omega} \alpha$$

As $\wedge_k(\mathbb{R}^n)$ and $\wedge_{n-k}(\mathbb{R}^n)$ have the same dimension there exists some isomorphisms between them. One of them is the Hodge star $*$ given by:

$$\alpha(\partial_1, ..., \partial_k) = (*\alpha)(\partial_{k+1}, ..., \partial_n)$$

when $\partial_1, ..., \partial_n$ is an oriented orthonormal basis of $\mathbb{R}^n$. The Hodge star simply transforms the basis ∂^I into $\partial^{\bar{I}}$ where $\bar{I} = \{1 \le i \le n | i \notin I\}$. On any n-dimensional manifold without border $\mathscr{S}$, the Hodge star implies an inner product on k-forms:

$$< \alpha, \beta >= \int_{\mathscr{S}} \alpha \wedge *\beta$$

Hodge star and exterior derivative allow to define the codifferential $\delta = *d* : \wedge_{k+1}(\mathbb{R}^n) \to \wedge_k(\mathbb{R}^n)$ which is the adjoint of d for $< ., . >$:

$$< d\alpha, \beta >=< \alpha, \delta\beta >$$

B The Mass and the Stiffness matrix

This appendix derives the expressions for the coefficients of the stiffness matrix Q and the mass matrix B. To do so, we start by parameterizing a triangle $t = (i, j, k)$ by the barycentric coordinates (or hat functions) Φ_i and Φ_j of a point $P \in t$ relative to vertices i and j. This allows to write $P = k + \Phi_i \mathbf{e}_j - \Phi_j \mathbf{e}_i$ (Figure 12). This yields an area element $dA(P) = \mathbf{e}_i \wedge \mathbf{e}_j d\Phi_i d\Phi_j = 2|t| d\Phi_i d\Phi_j$, where $|t|$ is the area of t, so we get the integral:

$$\int_{P \in t} \Phi_i \Phi_j dA = 2|t| \int_{\Phi_i=0}^{1} \int_{\Phi_j=0}^{1-\Phi_i} \Phi_i \Phi_j d\Phi_i d\Phi_j =$$

[4] In Einstein notation, an expression is implicitly summed over an index when it appears both on index and exposant

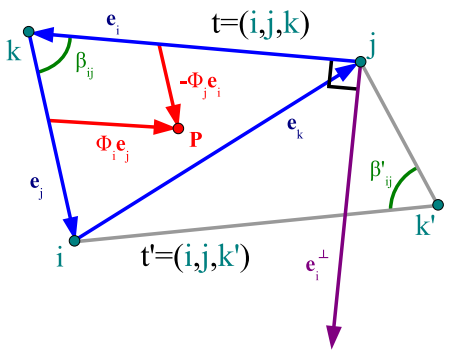

Figure 12: *Notations for matrix coefficients computation. Vectors are typed in bold letters* ($\mathbf{e}_i$)

$$|t| \int_{\Phi_i=0}^{1} \Phi_i (1-\Phi_i)^2 d\Phi_i = |t| \left(\frac{1}{2} - \frac{2}{3} + \frac{1}{4} \right) = \frac{|t|}{12}$$

which we sum up on the 2 triangles sharing (i, j) to get $B_{i,j} = (|t| + |t'|)/12$. We get the diagonal terms by:

$$\int_{P \in t} \Phi_i^2 dA = 2|t| \int_{\Phi_i=0}^{1} \int_{\Phi_j=0}^{1-\Phi_i} \Phi_i^2 d\Phi_j =$$

$$2|t| \int_{\Phi_i=0}^{1} \Phi_i^2 (1-\Phi_i) d\Phi_i = 2|t| \left(\frac{1}{3} - \frac{1}{4} \right) = \frac{|t|}{6}$$

which are summed up over the set $St(i)$ of triangles containing i to get $B_{i,i} = (\sum_{t \in St(i)} |t|)/6$.

To compute the coefficients of the stiffness matrix Q, we use the fact that d and δ are adjoint to get the more symmetric expression:

$$Q_{i,j} =< \Delta\Phi^i, \Phi^j >=< \delta d\Phi^i, \Phi^j >=< d\Phi^i, d\Phi^j >= \int_{\mathscr{S}} \nabla\Phi^i . \nabla\Phi^j$$

In t, the gradients of barycentric coordinates are the constants :

$$\nabla\Phi^i = \frac{-\mathbf{e}_i^\perp}{2|t|} \qquad \nabla\Phi^i . \nabla\Phi^j = \frac{\mathbf{e}_i . \mathbf{e}_j}{4|t|^2}$$

Where $\mathbf{e}_i^\perp$ denotes $\mathbf{e}_i$ rotated by $\pi/2$ around t's normal. By integrating on t we get:

$$\int_t \nabla\Phi^i . \nabla\Phi^j dA = \frac{\mathbf{e}_i . \mathbf{e}_j}{4|t|} = \frac{||\mathbf{e}_i|| . ||\mathbf{e}_j|| \cos(\beta_{ij})}{2||\mathbf{e}_i|| . ||\mathbf{e}_j|| \sin(\beta_{ij})} = \frac{\cot(\beta_{ij})}{2}$$

Summing these expressions, the coefficients of the stiffness matrix Q are given by:

$$Q_{i,i} = \sum_{t \in St(i)} \nabla\Phi^i . \nabla\Phi^i = \sum_{t \in St(i)} \frac{\mathbf{e}_i^2}{4|t|}$$

$$Q_{i,j} = \int_{t \cup t'} \nabla\Phi^i . \nabla\Phi^j = \frac{1}{2} \left(\cot(\beta_{ij}) + \cot(\beta'_{ij}) \right)$$

Note that this expression is equivalent to the numerator of the classical cotan weights.