Spectral Geometry Processing with Manifold Harmonics

B. Vallet\textsuperscript{1} and B. Lévy\textsuperscript{1}

\textsuperscript{1}ALICE team, INRIA

Abstract
We present an explicit method to compute a generalization of the Fourier Transform on a mesh. It is well known that the eigenfunctions of the Laplace Beltrami operator (Manifold Harmonics) define a function basis allowing for such a transform. However, computing even just a few eigenvectors is out of reach for meshes with more than a few thousand vertices, and storing these eigenvectors is prohibitive for large meshes. To overcome these limitations, 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. We also propose a limited-memory filtering algorithm, that does not need to store the eigenvectors. Using this latter algorithm, specific frequency bands can be filtered, without needing to compute the entire spectrum. Finally, we demonstrate some applications of our method to interactive convolution geometry filtering. These technical achievements are supported by a solid yet simple theoretic framework based on Discrete Exterior Calculus (DEC). In particular, the issues of symmetry and discretization of the operator are considered with great care.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, Hierarchy and geometric transformations

1. 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 (spatial frequencies). In his seminal paper, Taubin [Tauf] 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. He used this analogy as a theoretical tool to design and analyze approximations of low-pass filters. Several variants of this approach were then suggested, as discussed below.

In this paper, instead of only using Fourier analysis as a theoretical tool to analyze approximations of filters, our idea is to explicitly implement its generalization to surfaces of arbitrary topology, and use this to achieve interactive general convolution filtering. Our processing pipeline is similar to the one described in [PG01] for point sets. The main difference is that no resampling nor segmentation is needed: the Fourier transform is computed directly on the mesh. The 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 \( \omega_k \) denotes the frequency associated with \( H^k \);
- D: finally, transform the object back into geometric space by applying the inverse MHT.
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.

Note that this approach shares some similarities with signal processing with spherical harmonics or with the Discrete Cosine Transform. The main difference is that our approach does not require any resampling of the original surface. However, our MHB depends on the surface, and therefore requires to be precomputed (step A). 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
The main contribution of this paper is an efficient numerical mechanism to compute the eigenfunctions of the Laplacian that overcomes the current limits (thousands vertices) by several orders of magnitude (up to a million vertices). The eigenfunctions are computed band by band based on spectral transforms and an efficient eigensolver, and an out-of-core implementation that can compute thousands of eigenvectors for meshes with up to a million vertices. We also propose a limited-memory filtering algorithm, that does not need to store the eigenvectors. Once computed, the MHB allows for interactive filtering on meshes which we demonstrate applied to geometry. These computations use a clean, rigorous yet simple DEC-based framework which clarifies the issues of weighting and symmetry of the discrete cotangent Laplacian. In particular, the symmetry guarantees that the eigenfunctions form an orthonormal "Manifold Harmonics Basis" (MHB) with positive eigenvalues.

Previous Work
The discretization of the Laplacian operator plays a central role in geometry processing and has been extensively studied, motivated by the large number of its applications, that include parameterization, remeshing, compression, reconstruction and minimal surfaces. The eigenfunctions of the Laplacian are known to define a Fourier-like function basis $[ZvKD07]$, that can be used to define spectral analysis on manifolds. We will now review some of the most significant definitions and applications of discrete Laplacians, focusing mainly on the works related with spectral analysis and geometric filtering.

Combinatorial Laplacians: A combinatorial Laplacian solely depends on the connectivity of the mesh. Spectral analysis of the combinatorial graph Laplacian was first used by Taubin $[Tau95]$ to approximate low pass filters. A basis of eigenfunctions of the graph Laplacian was used by Karni et al. $[KG00]$ for geometry compression. Zhang $[Zha04]$ studies several variants of combinatorial laplacians and their properties for spectral geometry processing and JPEG-like mesh compression. However, as pointed-out in $[MDSB03]$, the analogy between the graph Laplacian and the discrete cosine transform supposes a uniform sampling of the mesh. Moreover, different embeddings of the same graph yield the same eigenfunctions, and two different meshings of the same object yield different eigenfunctions which is problematic as shown in Figure 2.

Geometric Laplacians and applications: More geometry can be injected into the definition of a discrete Laplacian through the ubiquitous cotan weights $[PP93,MDSB03]$. These weights can also be derived from Finite Element Modeling (FEM) such as done in $[WBH07]$, and they converge to the continuous Laplacian under certain conditions as explained in $[HPW06]$ and $[AFW06]$. Reuter et al. $[RWP05]$ also use FEM to compute the spectrum (i.e. the eigenva-
allow for correct reconstruction (C). Only our symmetric weights (geometric filtering method was recently proposed [A]) Laplacian-based Geometric Filtering: properties. definitions for a discrete Laplacian, depending on the desired exist on general meshes. This explains the large number of that satisfies all the properties of the continuous one cannot not allow for correct reconstruction (C). Only our symmetric weights (cot(βij) + cot(β′ij))/(A_i + A_j) are fully mesh-independent (E).

Figure 2: Filtering an irregularly sampled surface (twice denser on the right half) with different discrete laplacians. Combinatorial Laplacian (A), unweighted cotan cot(βij) + cot(β′ij) (B) and symmetrized weighted cotan (cot(βij) + cot(β′ij))/(A_i + A_j) (D) produce deformed results (right leg is bigger). Weighted cotan (cot(βij) + cot(β′ij))/A_i are not symmetric which does not need to resample the geometry, with the exception of Mousa et al. [MCA06] who directly compute the Spherical Harmonic Transform of a star-shaped mesh. It is also possible to extract the frequencies from a progressive mesh [LSS*98] and avoid resampling the geometry by using irregular subdivision [GSS99]. Finally, an original approach is presented in [SCOIT05] that computes "geometry aware" basis functions, defined as solutions of some least-squares problems. Our method computes the frequency-space basis functions as eigenvectors of a symmetric matrix, for a surface of arbitrary topology without needing any resampling or segmentation.

The rest of the paper is organized as follows. We first recall some notions on the Fourier Analysis (Section 2.1). A Manifold Harmonics Basis (MHB) is then built through a DEC formulation, and its relations with the classical discrete Laplacian is explained (Section 2.2). Equipped with this new tool, it is then simple to generalize spectral analysis and Fourier transform to arbitrary manifolds. This defines the Manifold Harmonics Transform (MHT) that transforms from geometric space into frequency space, and the inverse MHT (Section 3). We finally explain how to efficiently compute the MHB in practice, and how to implement scalable approaches to geometric filtering are based on energy minimization (e.g. band-exaggeration). This method combines explicit and implicit schemes to reach the different frequency bands involved in the filter. In comparison, our method can use arbitrary user-defined filters, and offers in addition the possibility of changing the filter interactively.

Geometric filtering through energy minimization: Other approaches to geometric filtering are based on energy minimization (e.g. [Mal92]). These methods are called discrete fairing in [Kot97, KCVS98], in reference to their continuous-setting counterparts [BW90]. Recently, a method was proposed [NISA06] 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.

Spectral transform: To directly implement the spectral transform on manifolds, several methods consist in putting the input surface in one-to-one correspondence with a simpler domain [ZBS04], or to partition it into a set of simpler domains [LSS*98, PG01] 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. [MCA06] who directly compute the spherical harmonic transform of a star-shaped mesh. It is also possible to extract the frequencies from a progressive mesh [LSS*98] and avoid resampling the geometry by using irregular subdivision [GSS99]. Finally, an original approach is presented in [SCOIT05] that computes "geometry aware" basis functions, defined as solutions of some least-squares problems. Our method computes the frequency-space basis functions as eigenvectors of a symmetric matrix, for a surface of arbitrary topology without needing any resampling or segmentation.

The rest of the paper is organized as follows. We first recall some notions on the Fourier Analysis (Section 2.1). A Manifold Harmonics Basis (MHB) is then built through a DEC formulation, and its relations with the classical discrete Laplacian is explained (Section 2.2). Equipped with this new tool, it is then simple to generalize spectral analysis and Fourier transform to arbitrary manifolds. This defines the Manifold Harmonics Transform (MHT) that transforms from geometric space into frequency space, and the inverse MHT (Section 3). We finally explain how to efficiently compute the MHB in practice, and how to implement scalable approaches to geometric filtering are based on energy minimization (e.g. band-exaggeration). This method combines explicit and implicit schemes to reach the different frequency bands involved in the filter. In comparison, our method can use arbitrary user-defined filters, and offers in addition the possibility of changing the filter interactively.
spectral geometry processing (Section 4). We conclude by presenting some applications and results.

2. Spectral Analysis on Manifolds

Manifold harmonics (also called shape harmonics) are defined as the eigenfunctions of the Laplace operator. This section starts by recalling the familiar Fourier analysis to justify the choice of the eigenfunctions of the Laplace operator to generalize this setting to arbitrary manifolds. We will then define the Laplace operator in the DEC setting and justify a symmetric weighting ensuring the orthogonality of the MHB.

2.1. Fourier Analysis

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:

\[
f(x) = \sum_{k=1}^{\infty} \hat{f}_k H^k(x) ; \quad \left\{ \begin{array}{l} H_{2k}^k = \sin(kx) \\ H_{2k+1}^k = \cos(kx) \end{array} \right.
\]

(1)

where the coefficients \( \hat{f}_k \) of the decomposition are given by:

\[
\hat{f}_k = \langle f, H^k \rangle = \int_0^1 f(x) H^k(x) dx
\]

(2)

and where \( \langle ..., \rangle \) denotes the inner product (i.e. the “dot product” for functions defined on \([0, 1]\)). The “Circle harmonics” basis \( H^k \) is orthonormal with respect to \( \langle ..., \rangle \):

\[
\langle H^k, H^l \rangle > 0 \text{ if } k \neq l,
\]

The set of coefficients \( \hat{f}_k \) (Equation 2) is called the Fourier Transform (FT) of the function \( f \). Given the coefficients \( \hat{f}_k \), 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 \( H_{2k}^k \) (resp. \( H_{2k+1}^k \)) are associated with the eigenvalues \( k^2 \):

\[
-\frac{\partial^2 H_{2k}^k(x)}{\partial x^2} = k^2 \sin(kx) = k^2 H_{2k}^k(x)
\]

To extended this construction to arbitrary manifolds, we need to generalize both inner product \( \langle ..., \rangle \) and second-order derivative \( \partial^2/\partial x^2 \) to arbitrary manifolds. We now explain how Discrete Exterior Calculus (DEC) helps doing so.

2.2. DEC for spectral analysis

For a complete introduction to DEC we refer the reader to [DKT05], [Hir03] and to [AFW06] for proofs of convergence. We quickly introduce the few notions and notations that we are using to define the inner product \( \langle ..., \rangle \) and generalized second-order derivative (i.e. Laplacian operator).

A \( k \)-simplex \( s_k \) is the geometric span of \( k + 1 \) points. For instance, 0, 1 and 2-simplices are points, edges and triangles respectively. In our context, a mesh can be defined as a 2-dimensional simplicial complex \( S \), i.e. a collection of \( n_k \) \( k \)-simplices \((k = 0, 1, 2)\), with some conditions to make it manifold. A discrete \( k \)-form \( \omega^k \) on \( S \) is given by a real value \( \omega^k(s_k) \) associated with each oriented \( k \)-simplex (that corresponds to the integral of a smooth \( k \)-form over the \( k \)-simplex). The set \( \mathcal{Q}^k(S) \) of \( k \)-forms on \( S \) is a vector space of dimension \( n_k \). With a proper numbering of the \( k \)-simplices, \( \omega^k \) can be assimilated to a vector of size \( n_k \) and linear operators from \( \mathcal{Q}^k(S) \) to \( \mathcal{Q}^l(S) \) can be assimilated to \((n_k, n_l)\) matrices.

The exterior derivative \( d_k : \mathcal{Q}^k(S) \rightarrow \mathcal{Q}^{k+1}(S) \) is defined by the signed adjacency matrix: \((d_k)_{s_k,s_{k+1}} = \pm 1 \) if \( s_k \) belongs to the boundary of \( s_{k+1} \), with the sign depending on their respective orientations.

DEC provides \( \mathcal{Q}^k(S) \) with a \( L_2 \) inner product:

\[
\langle \omega^k_1, \omega^k_2 \rangle > = (\omega^k_1)\,^T \omega^k_2
\]

(3)

where \( s_k \) is the so-called Hodge star. As a matrix, the Hodge star is diagonal with elements \(|s_k^2|/|s_k|\) where \( s_k^2 \) denotes the circuncentric dual of simplex \( s_k \) and \(|.|\) is the simplex volume. In particular, for vertices, edges and triangles:

\[
\langle \ast \circ v, \circ v' \rangle > = |v'| : \langle \ast \circ e, \circ e' \rangle > > = \frac{|e'|}{|e|} \cot \beta_e + \cot \beta'_e
\]

where \( \beta_e \) and \( \beta'_e \) denote the two angles opposite to \( e \).

Finally the Laplace of Rham operator on 0-forms is given by:

\[
\Delta = -s_0^{-1} d_0^T d_0 d_0^T \ast : \Delta_{ij} = -\sum_j \Delta_{ij}
\]

For surfaces with borders, if the edge \( ij \) is on the border, the term \( \cot(\beta_{ij}) \) vanishes and the dual cell \( s_i^2 \) is cropped by the border. This matches the FEM formulation (see [HL88] or [VL07]) with Neumann boundary conditions, with the difference that FEM has a mass matrix (corresponding to \( s_0 \)) which is not diagonal and which terms consist of one ring areas (whereas \( s_0 \) contains dual cell areas). This comes from the fact that DEC is based on a mixed Finite Elements/Finite Volumes scheme where the elements are the same as in FEM but the volumes are the dual cells. This difference has little impact, especially as the mass matrix is often diagonalized (lumped mass approximation) in FEM formulations.

Remark: The matrix \( \Delta \) corresponds to the standard discrete Laplacian, except for the sign. The sign difference comes from the distinction between Laplace-Beltrami and Laplace de Rham operators.

The so-defined Laplacian \( \Delta \) apparently looses the symmetry of its continuous counterpart \((\Delta_{ij} \neq \Delta_{ji})\). This makes the
eigenfunction basis no longer orthonormal, which is problematic for our spectral processing purposes (Figure 2-C). To recover symmetry, consider the canonical basis $\{\phi_i\}$ of 0-forms: $\phi_i$ is 1 on vertex $i$ and $\phi_i = 0$ on other vertices. This basis is orthogonal but not normal with respect to the inner product defined in Equation 3 \(<\phi_i, \phi_j> = (\phi_i)^T \phi_j \neq 1\). However, since the Hodge star $\star_0$ is a diagonal matrix, one can easily normalize $\{\phi_i\}$ as follows:

$$\hat{\phi}_i = \frac{1}{\sqrt{\sum_j |v_j|^2}} \phi_i$$

In this orthonormalized basis $\{\hat{\phi}_i\}$, the Laplacian $\hat{\Delta}$ is symmetric, and its coefficients are given by:

$$\hat{\Delta} = \frac{1}{2} \Delta \sigma_0^{-1/2} \quad \hat{\Delta}_{ij} = -\frac{\cot \beta_{ij} + \cot \beta_{ij}'}{\sqrt{|v_i|^2 |v_j|^2}}$$  \hspace{1cm} (4)

2.3. The Manifold Harmonic Basis (MHB)

Given these definitions, we can now compute the MHB, defined to be the set of eigenvectors of $\hat{\Delta}$ expressed in the canonical basis:

1. Assemble the discrete Laplacian $\hat{\Delta}$ (see Equation 4);
2. Compute its eigenvectors $(\hat{H}^k)$ (see Section 4);
3. Map them into the canonical basis: $(\bar{H}^k) = (\sigma_0^{-1/2} \bar{H}^k)$.

The set of so-obtained $(\bar{H}^k)$ vectors is called the Manifold Harmonics basis (MHB). An example is shown in Figure 3. Note that by construction, the MHB is orthonormal with respect to the inner product. This property is important to define the associated transforms, as explained below.

3. The Manifold Harmonic Transform

Transforming 0-forms between the canonical basis $\{\phi_i\}$ i.e., geometric space and the MHB $(\bar{H}^k)$ i.e., frequency space will be called the Manifold Harmonic Transform (MHT). It is also easy to define the inverse transform (MHT$^{-1}$). They generalize the notion of Fourier transform to simplicial complexes. We now derive the expressions of MHT and MHT$^{-1}$, and apply them to geometry filtering.

3.1. MHT and inverse MHT

We consider the geometry $x$ (resp. $y, z$) of the simplicial complex $\Sigma$, that can be seen as a piecewise linear function. In other words, $x$ is a linear combination of the canonical basis functions $\phi^i$:

$$x = \sum_{i=1}^n x_i \phi^i$$  \hspace{1cm} (5)

where $x_i$ denotes the $x$ coordinate at vertex $i$. Computing the MHT means projecting the function $x$ into the MHB, i.e. finding the coefficients $\tilde{x}_k$ such that:

$$x = \sum_{k=1}^m \tilde{x}_k \bar{H}^k$$  \hspace{1cm} (6)

Since the MHB is orthonormal, we can easily project (5) and (6) onto each $\bar{H}^k$ with the inner product:

$$\langle x, \bar{H}^k \rangle = \sum_{i=1}^n x_i \phi^i \langle \phi^i, \bar{H}^k \rangle = x^T \sigma_0 \bar{H}^k$$

$$\langle x, \bar{H}^k \rangle = \sum_{k=1}^n \tilde{x}_k \langle \bar{H}^k, \bar{H}^k \rangle = \tilde{x}_k$$

By equating both expressions, we finally obtain the expression of $\tilde{x}_k$:

$$\tilde{x}_k = x^T \sigma_0 \bar{H}^k = \sum_{i=1}^n x_i \langle \phi^i, \bar{H}^k \rangle$$  \hspace{1cm} (7)

Using these equations, we know how to transform 0-forms between geometric space $(x)$ and frequency space $(\tilde{x})$. This defines the MHT and inverse MHT:

- MHT: $(x) \rightarrow (\tilde{x})$ (Equation 7)
- MHT$^{-1}$: $(\tilde{x}) \rightarrow (x)$ (Equation 6)

Figure 4 shows the geometry reconstructed from the MHT of a surface using a different number $m$ of MHT coefficients. As can be seen, the first $H^k$ functions capture the general shape of the functions and the next ones correspond to the details.

3.2. Filtering

Once the geometry is converted into the MHB, each component $(\tilde{x}_k, \tilde{\phi}_k)$ of the MHT corresponds to an individual spatial frequency $\omega_k$. In the case of a closed curve (Section
demonstrates low-pass, enhancement and band-
secondly, we need to compute a large number of eigenvec-
into the inverse MHT, as follows:
The frequency-space filter can be applied to the high-
ter the nodal sets \[ \text{Equation (8)} \]
does not need to re-scale the model. Moreover, only the fil-
exaggeration filters. Note that unlike existing methods, ours
equality filters can be obtained, that connect \( \omega_k = \sqrt{\lambda_k} \) with the length of the
In practice, we stop computing the MHB at a given “cutoff”
for the other end of the spectrum. This can be explained
computations are known to be extremely computationally intensive. To reduce
the Arnoldi method. Yet, two characteristics of
eigenvector solvers hinder us from using them directly to
To compute the solutions of large sparse eigenproblems, sev-
the publicly available library ARPACK (used in [DBG’06]) provides an efficient imple-
for meshes with more than a few thousand vertices [WK05]. In this section, we show how this limit can be
overcome by several orders of magnitude.

4. Numerical Solution Mechanism

Computing the MHB means solving for the eigenvalues \( \lambda_k \) and eigenvectors \( \vec{H}_k \) for the symmetric positive semi-definite matrix \( \Delta \):
\[
\bar{\Delta} \vec{H}_k = \lambda_k \vec{H}_k \quad (9)
\]
However, eigenvalues and eigenvectors computations are
computed, and it is well known that computation time is superlinear in the number of requested
Figure 4: Reconstructions obtained with an increasing number of MH functions.

2.1), we have \( -\partial^2 \sin(\omega x) / \partial x^2 = \omega^2 \sin(\omega x) \). Therefore, the spatial frequency \( \omega_0 \) corresponds to the square root of the eigenvalue \( \lambda_4 \). In the case of a manifold, similar results can be obtained, that connect \( \omega_k = \sqrt{\lambda_k} \) with the length of the nodal sets [DF88].

A frequency-space filter is a function \( F(\omega) \) that gives the amplification to apply to each spatial frequency \( \omega \). Since all frequencies are separated by the MHT, applying a filter \( F(\omega) \) to the geometry becomes a simple product in frequency space. The filtered coordinate \( x'_i \) (resp. \( y'_i, z'_i \)) at vertex \( i \) is then given by:
\[
x'_i = \sum_{k=1}^{m} F(\omega_k) \tilde{x}_k H^k_i = \sum_{k=1}^{m} F(\sqrt{\lambda_k}) \tilde{x}_k H^k_i
\]
In practice, we stop computing the MHB at a given “cutoff”
for the other end of the spectrum. This can be explained
computations are known to be extremely computationally intensive. To reduce
the Arnoldi method. Yet, two characteristics of
eigenvector solvers hinder us from using them directly to
To compute the solutions of large sparse eigenproblems, sev-
the publicly available library ARPACK (used in [DBG’06]) provides an efficient imple-
for meshes with more than a few thousand vertices [WK05]. In this section, we show how this limit can be
overcome by several orders of magnitude.

4. Numerical Solution Mechanism

Computing the MHB means solving for the eigenvalues \( \lambda_k \) and eigenvectors \( \vec{H}_k \) for the symmetric positive semi-definite matrix \( \Delta \):
\[
\bar{\Delta} \vec{H}_k = \lambda_k \vec{H}_k \quad (9)
\]
However, eigenvalues and eigenvectors computations are known to be extremely computationally intensive. To reduce
the computation time, Karni et al. [KG00] partition the mesh into smaller charts, and [DBG’06] use multiresolution tech-
in our case, we need to compute multiple eigenvectors (typically a few thousands). This is known to be cur-
rently impossible for meshes with more than a few thousand vertices [WK05]. In this section, we show how this limit can be
overcome by several orders of magnitude.

To compute the solutions of large sparse eigenproblems, sev-
several iterative algorithms exist. The publicly available library ARPACK (used in [DBG’06]) provides an efficient imple-
the Arnoldi method. Yet, two characteristics of
eigenvector solvers hinder us from using them directly to
calculate 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 \( \omega_m \). Unfortunately, iterative solvers perform much better for the other end of the spectrum. 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

Figure 5 demonstrates low-pass, enhancement and band-
ne is then given by:
\[
x'_i = \sum_{k=1}^{m} F(\omega_k) \tilde{x}_k H^k_i = \sum_{k=1}^{m} \sqrt{\lambda_k} \tilde{x}_k H^k_i
\]
In practice, we stop computing the MHB at a given “cutoff”
for the other end of the spectrum. This can be explained
computations are known to be extremely computationally intensive. To reduce
the computation time, Karni et al. [KG00] partition the mesh into smaller charts, and [DBG’06] use multiresolution tech-
in our case, we need to compute multiple eigenvectors (typically a few thousands). This is known to be cur-
rently impossible for meshes with more than a few thousand vertices [WK05]. In this section, we show how this limit can be
overcome by several orders of magnitude.

To compute the solutions of large sparse eigenproblems, sev-
several iterative algorithms exist. The publicly available library ARPACK (used in [DBG’06]) provides an efficient imple-
the Arnoldi method. Yet, two characteristics of
eigenvector solvers hinder us from using them directly to
calculate 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 \( \omega_m \). Unfortunately, iterative solvers perform much better for the other end of the spectrum. 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
eigenvectors. In addition, if the surface is large (a million vertices), the MHB does not fit in system RAM.

4.1. Band-by-band computation of the MHB

We address both issues by applying spectral transforms to the eigenproblem. To get the eigenvectors of a spectral band centered around a value $\lambda_S$, we start by shifting the spectrum by $\bar{\lambda}_S$, by replacing $\Delta$ with $\Delta = \bar{\lambda} = \lambda_S \mathbf{I}$. Then, we can swap the spectrum by inverting $\Delta_S$. This is called the Shift-Invert spectral transform, and the new eigenproblem to solve is given by:

$$\Delta_S^{-1} \bar{H}^k = \mu_k \bar{H}^k \quad \text{(10)}$$

The fact that $\Delta_S$ may be singular is in practice not a problem as the iterative solver we use to solve (10) does not ask for $\Delta_S^{-1}$ explicitly. Instead, it outputs some vectors $\vec{v}$ and asks in return for the result of the multiplication $\vec{x} = \Delta_S^{-1} \vec{v}$. Hence we only need to compute a sparse indefinite Cholesky factorization of $\Delta_S$ using sparse direct solvers (TAUCS, SUPERLU). Thanks to this factorization, it is very fast to solve $\Delta_S \vec{x} = \vec{v}$ by simple back-substitution each time the iterative solver asks for a multiplication with $\Delta_S^{-1}$. For large models (a million vertices), we used the sparse OOC (out-of-core) symmetric indefinite factorization [MIT06] implemented in the future release of TAUCS, kindly provided by S. Toledo.

It is easy to check that (9) and (10) yield the same eigenvectors, and that the eigenvalues are related by $\lambda_k = \bar{\lambda} + 1/\mu_k$. Our iterative solver returns the high end of the spectrum (largest $\mu$’s), corresponding to a band of eigenvalues of $\bar{\lambda}$ centered around $\bar{\lambda}_S$. 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. Based on the considerations above, we derive the following “band-by-band” algorithm:

$$\begin{align*}
(1) \quad & \lambda_S \leftarrow 0 \quad ; \quad \lambda_{\text{start}} \leftarrow 0 \\
(2) \quad & \textbf{while} (\lambda_{\text{start}} < \bar{\lambda}_S^{\text{end}}) \\
(3) \quad & \text{compute the Cholesky factorization of } \Delta_S = \bar{\lambda} - \lambda_S \mathbf{I} \\
(4) \quad & \text{find the 50 first eigenpairs } (\bar{H}^k, \mu_k) \text{ of } \Delta_S^{-1} \\
(5) \quad & \textbf{for } k = 1 \text{ to } 50 \\
(6) \quad & \lambda_k \leftarrow \lambda_S + 1/\mu_k \\
(7) \quad & \textbf{if } (\lambda_k > \lambda_{\text{start}}) \textbf{ write}(\bar{H}^k, \lambda_k) \\
(8) \quad & \textbf{end } / / \textbf{for} \\
(9) \quad & \lambda_{\text{start}} \leftarrow \max(\lambda_k) + 0.4(\max(\lambda_k) - \min(\lambda_k)) \\
(10) \quad & \bar{\lambda}_{\text{start}} \leftarrow \max(\lambda_k) \\
(11) \quad & \textbf{end } / / \textbf{while}
\end{align*}$$

If the MHB does not fit in RAM, the new eigenpairs of each band of can be stream-written into a file (Line 7). Since the eigenvalues are centered around the shift $\lambda_S$, 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). If the bands do not overlap, we recompute a larger band until they do. Note that this is different from the shift-invert spectral transform implemented by ARPACK, dedicated to iterative solvers. Ours makes use of the factorization of the matrix, resulting in much better performances.

4.2. Limited-memory MH Filtering

After computing the MHB (Section 4.1) the filtering algorithm (Section 3.2) allows for interactive spectral geometry processing. However, for large meshes, storing the MHB may be prohibitive. Fortunately, it is possible to compute the MHB and the filter all together (at the cost of loosing interactivity). In addition, computations can be reordered in a way that requires storing only one eigenvector at a time:

$$\begin{align*}
(1) \quad & x^F \leftarrow x \\
(2) \quad & \textbf{for each eigenpair } (H^k, \omega_k) \\
(3) \quad & x^F \leftarrow x^F + (F(\omega_k) - 1) <x,H^k> H^k \\
& \quad \text{(where } <x,H^k> = x^T \omega_k H^k) \\
(4) \quad & \textbf{end for}
\end{align*}$$

Figure 5: Low-pass, enhancement and band-exaggeration filters. The filter can be changed by the user, the surface is updated interactively.
In a certain sense, this algorithm replaces a given frequency in the mesh by its filtered version. Once again, this is made possible by the orthonormality of the MHB. In practice, the combined MHB+filtering algorithm is obtained by replacing line (7) of the MHB computation algorithm with line (3) of the algorithm above (instead of streaming the eigenpair to a file, we replace the corresponding frequency in the mesh). Interestingly, since our limited-memory algorithm changes a specific frequency in-place (i.e. without storing the MHB), it is possible to limit the computation of the MHB to the frequencies changed by the filter, by using the shift-invert spectral transform.

5. Results, Discussion and Conclusions

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.

We have experimented our filtering method with objects of different sizes. The timings are reported in Table 1. The last row (limited-memory version) uses a band-exaggeration filter that spans 1/4th of the object’s spectrum.

Table 1: Timings for the different phases of the algorithm.

<table>
<thead>
<tr>
<th></th>
<th>dino</th>
<th>drago</th>
<th>drago1</th>
<th>drago2</th>
<th>drago3</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>56K</td>
<td>150K</td>
<td>244K</td>
<td>500K</td>
<td>1M</td>
</tr>
<tr>
<td>m</td>
<td>447</td>
<td>315</td>
<td>667</td>
<td>800</td>
<td>1331</td>
</tr>
<tr>
<td>MHB</td>
<td>77 s</td>
<td>160 s</td>
<td>9 m</td>
<td>2 h 21 m</td>
<td>6 h</td>
</tr>
<tr>
<td>MHT</td>
<td>0.34 s</td>
<td>0.65 s</td>
<td>18 s</td>
<td>32 s</td>
<td>76 s</td>
</tr>
<tr>
<td>MHT−1</td>
<td>0.53 s</td>
<td>1.02 s</td>
<td>4 s</td>
<td>48 s</td>
<td>85 s</td>
</tr>
<tr>
<td>LM-filt</td>
<td>18 s</td>
<td>41 s</td>
<td>135 s</td>
<td>28 m</td>
<td>1h</td>
</tr>
</tbody>
</table>

In this paper, we have presented a method for filtering functions defined on manifolds. Our method and associated numerical solution mechanism may find applications in various contexts, e.g. segmentation, mesh watermarking or reconstruction. We have explained how a DEC framework allows to recover the symmetry of discrete Laplace operators and the orthogonality of its eigenvectors. We used this theoretical framework to define the Manifold harmonic transform and inverse. After precomputing the MHB, the resulting filter is interactive and does not encounter the shrinking effect obtained with classical schemes. On the practical side, we have overcome the current size limits of spectral geometry processing by several orders of magnitudes, by making it usable for meshes with up to $10^5 \sim 10^9$ vertices. With our limited-memory MH filtering algorithm, storage space is no longer a limit. However, processing time for the MHB starts to be expensive (hours) beyond $10^6$ vertices. This will be optimized in future works, by introducing multisresolution in our solution mechanism.

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. 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 solver applied to the uniform Laplacian can be also used of the spectrum (that corresponds to large geometric details). 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 [KR05]. In addition, since our filter explicitly implements the spectral transform, it does not introduce the shrinking effect usually encountered with existing ones.

Conclusions

Figure 7: Left: a sphere and a genus-4 model with random noise added. Right: the low-pass filtered result.
Figure 6: The MHB computed on 1M vertices (XYZ dragon) and limited-memory MH-filtering (that does not store the MHB).

Figure 8: Filtering Stanford’s bunny. Results similar to geofilter are obtained, with the addition of interactivity, and without any shrinking effect.

to implement an efficient version of Karni et al.’s Spectral Mesh Compression [KG00]. Our method cannot be directly applied to mesh compression as we took particular care making our Laplacian geometry dependant. In this case, a purely combinatorial Laplacian is required as in Karni et al.’s Spectral Mesh Compression [KG00]. Moreover, it turns out that because the MHB is not spatially localized, many MHT coefficients (several thousands) are required to accurately reconstruct the geometry, even with a geometry aware Laplacian (see Figure 4). Besides Karni’s initial concern of reducing computation time, we think that partitioning also partially fixes the problem of spatial localization (this is also why JPEG uses small blocks instead of applying the DCT to the whole image) at the expense of losing continuity. This leads to forecast that defining Manifold Wavelets localized in both frequency and spatial domains [GKS02] will be an exciting research avenue in the future.

Acknowledgments

We gratefully thank all the reviewers of this paper and especially the senior reviewer for helping us to enhance the quality of this paper. We wish to thank Matthias Zwicker for the suggestion of studying manifold learning, Ramsey Dyer for discussions about orthogonality, Sivan Toledo for providing us with his latest TAUCS package. We are also grateful to Michela Spagnulo for encouraging us to continue in this direction, and Michael Garland for discussions on the eigenfunctions. We thank Raphaëlle Chaine and Remi Allegrre for some of the mesh models. This work was funded by AIM@Shape (FP6 European Network of Excellence), ARC GEOREP (INRIA grant) and Geometric Intelligence (Microsoft Research grant).

References


[DKT05] Desbrun M., Kanzo E., Tong Y.: Dis-
 discrete differential forms for computational modeling. *Sig- 

graph ’05 course notes on Discrete Differential Geometry, 

Chapter 7* (2005).


