Spectral Mesh Representation

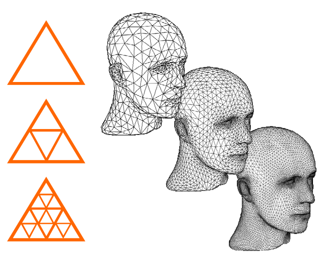
Slides from Olga Sorkine

Motivation

- Want frequency domain representation for 3D meshes
  - Smoothing
  - Compression
  - Progressive transmission
  - Watermarking
  - etc.

Frequencies in a mesh

- One possibility = multires meshes



[Hoppe]

Frequencies in a function

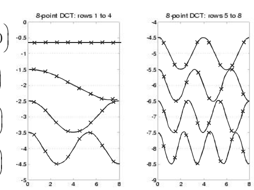
- Fourier analysis (DCT)
  - cosine bases for signals:

$\cos\left(\frac{0\cdot\pi}{16}(2x-1)\right)$

$\cos\left(\frac{\pi}{16}(2x-1)\right)$

$\cos\left(\frac{2\pi}{16}(2x-1)\right)$

$\cos\left(\frac{3\pi}{16}(2x-1)\right)$



$\cos\left(\frac{4\pi}{16}(2x-1)\right)$

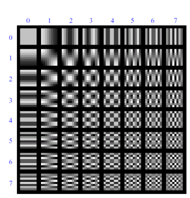
$\cos\left(\frac{5\pi}{16}(2x-1)\right)$

$\cos\left(\frac{6\pi}{16}(2x-1)\right)$

$\cos\left(\frac{7\pi}{16}(2x-1)\right)$

Frequencies in a function

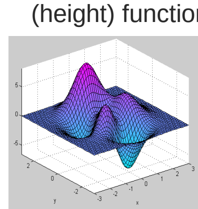
- Fourier analysis (DCT)
  - 2D bases for 2D signals (images)



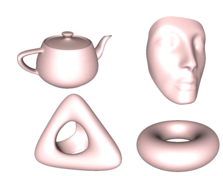
$\cos\left(\frac{\pi u}{16}(2x+1)\right)\cos\left(\frac{\pi v}{16}(2y+1)\right)$

How about 3D shapes?

- Problem: general 3D shapes are not (height) functions



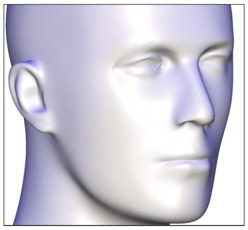
Height function, regularly sampled above a 2D domain



General 3D shapes

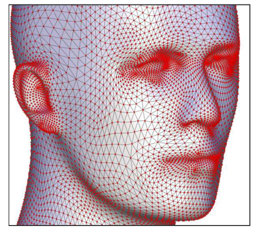
Irregular meshes

- In graphics, shapes are mostly represented by triangle meshes



Irregular meshes

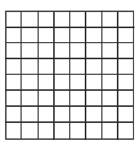
- In graphics, shapes are mostly represented by triangle meshes



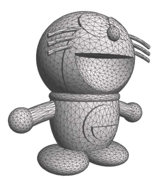
- Geometry: Vertex coordinates
  (x_1, y_1, z_1)
  (x_2, y_2, z_2)
  ...
  (x_n, y_n, z_n)
- Connectivity: List of triangles
  (i_1, j_1, k_1)
  (i_2, j_2, k_2)
  ...
  (i_m, j_m, k_m)

How to define efficient bases?

- Extension of the 2D DCT basis to a general (irregular) mesh



DCT

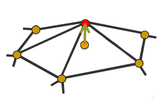


???

Basis functions for 3D meshes

- We need a collection of **basis functions**
  - First basis functions will be very smooth, slowly-varying
  - Last basis functions will be high-frequency, oscillating
- We will represent our shape (mesh geometry) as a **linear combination** of the basis functions
- We will induce the basis functions from the mesh **connectivity**!

The Mesh Laplacian operator



$$L(\mathbf{v}_i) = d_i \mathbf{v}_i - \sum_{j \in N(i)} \mathbf{v}_j = d_i \left(\mathbf{v}_i - \frac{1}{d_i} \sum_{j \in N(i)} \mathbf{v}_j \right)$$

- Measures the local smoothness at each mesh vertex

Laplacian operator in matrix form

$$\begin{pmatrix} d_1 & -1 & 0 & \cdots & -1 & \cdots & \cdots & 0 \\ 0 & d_2 & & -1 & & & -1 & \\ \vdots & & d_3 & & & & & \\ \vdots & & & \ddots & & & & \\ \vdots & & & & \ddots & & & \\ \vdots & & & & & \ddots & & \\ 0 & -1 & & -1 & & -1 & d_{n-1} & \\ -1 & & -1 & & -1 & & & d_n \end{pmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{v}_{n-1} \\ \mathbf{v}_n \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \delta_{n-1} \\ \delta_n \end{pmatrix}$$

L matrix

Spectral bases

- L is a symmetric n×n matrix
- Perform spectral analysis of L!

$$L = \begin{pmatrix} | & | & & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \\ | & | & & | \end{pmatrix} \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix} \begin{pmatrix} | & | & & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \\ | & | & & | \end{pmatrix}^T$$

Basis vectors

Frequencies, sorted in ascending order

How to represent our mesh geometry in the spectral basis?

- $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \in \mathbf{R}^n$. Decompose in the spectral basis:

$$\mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \alpha_1\mathbf{b}_1 + \alpha_2\mathbf{b}_2 + \ldots \alpha_n\mathbf{b}_n$$

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \beta_1\mathbf{b}_1 + \beta_2\mathbf{b}_2 + \ldots \beta_n\mathbf{b}_n$$

$$\mathbf{Z} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} = \gamma_1\mathbf{b}_1 + \gamma_2\mathbf{b}_2 + \ldots \gamma_n\mathbf{b}_n$$

How to represent our mesh geometry in the spectral basis?

- Decompose the mesh geometry in the spectral basis:

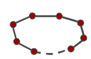
$$\begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_n \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix}^T \mathbf{b}_1 + \begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix}^T \mathbf{b}_2 + \ldots + \begin{pmatrix} \alpha_n \\ \beta_n \\ \gamma_n \end{pmatrix}^T \mathbf{b}_n$$

The first components are low-frequency

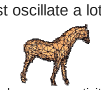
The last components are high-frequency

The spectral basis

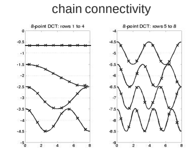
- First functions are smooth and slow, last oscillate a lot



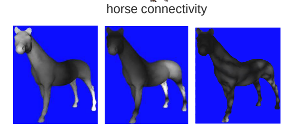
chain connectivity



horse connectivity



spectral basis of L = the DCT basis



2[nd] basis function

10[th] basis function

100[th] basis function

The spectral basis

- Low-frequency basis vectors are smooth and slowly-varying:
  - The first basis vector:

$$L\mathbf{b}_1 = 0$$
$$\mathbf{b}_1 = (1, 1, \ldots 1)^T$$

  - It is the smoothest possible function (Laplacian is constant **zero** on it)

The spectral basis

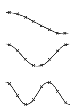
- Low-frequency basis vectors are smooth and slowly-varying:
  - The first basis vector:

$$L\mathbf{b}_1 = 0$$
$$\mathbf{b}_1 = (1, 1, \ldots 1)^T$$

  - $L\mathbf{b}_1 = 0$ because all the rows of L sum to zero

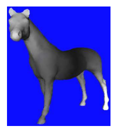
The spectral basis

- Low-frequency basis vectors are smooth and slowly-varying:
  - The following basis vectors:



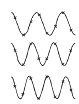
$$L\mathbf{b}_i = \lambda_i \mathbf{b}_i$$

The frequency is small → the Laplacian operator gives small values → the shape of the function is smooth



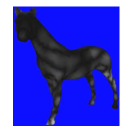
The spectral basis

- High-frequency basis vectors are non-smooth, oscillating:



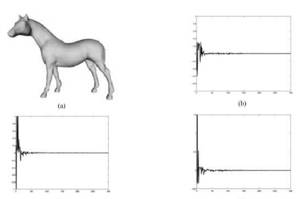
$$L\mathbf{b}_i = \lambda_i \mathbf{b}_i$$

The frequency is large → the Laplacian operator gives large values → the shape of the function is less smooth!



The spectral basis

- Most shape information is in low-frequency components



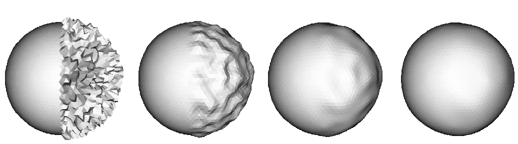
[Karni and Gotsman 00]

Applications

- Smoothing
- Compression
- Progressive transmission
- Watermarking
- etc.

Mesh smoothing

- Aim to remove high frequency details



[Taubin 95]

Spectral mesh smoothing

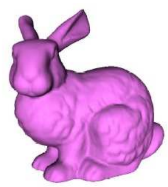
- Drop the high-frequency components

$$\begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_n \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{pmatrix}^{\mathrm{T}} \mathbf{b}_1 + \begin{pmatrix} \alpha_2 \\ \beta_2 \\ \gamma_2 \end{pmatrix}^{\mathrm{T}} \mathbf{b}_2 + \ldots + \begin{pmatrix} \alpha_n \\ \beta_n \\ \gamma_n \end{pmatrix}^{\mathrm{T}} \mathbf{b}_n$$

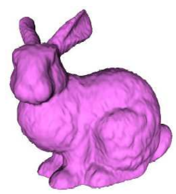
High-frequency components!

Mesh compression

- Aim to represent surface with fewer bits



36 bits/vertex



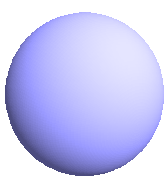
1.4 bits/vertex

Mesh compression

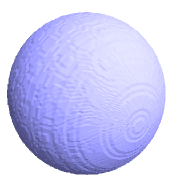
- Most of mesh data is in geometry
  - The connectivity (the graph) can be very efficiently encoded
    - About 2 bits per vertex only
  - The geometry (x,y,z) is heavy!
    - When stored naively, at least 12 bits per coordinate are needed, i.e. 36 bits per vertex

Mesh compression

- What happens if quantize xyz coordinates?



original



8 bits/coordinate

Mesh compression

- Quantization of the Cartesian coordinates introduces high-frequency errors to the surface.
- High-frequency errors alter the visual appearance of the surface – affect normals and lighting.

Mesh compression

- Transform the Cartesian coordinates to another space where quantization error will have low frequency in the regular Cartesian space
- Quantize the transformed coordinates.
- Low-frequency errors are less apparent to a human observer.

Spectral mesh compression

- The encoding side:
  - Compute the spectral bases from mesh connectivity
  - Represent the shape geometry in the spectral basis and decide how many coeffs. to leave (**K**)
  - Store the connectivity and the **K** non-zero coefficients
- The decoding side:
  - Compute the first **K** spectral bases from the connectivity
  - Combine them using the **K** received coefficients and get the shape

Why it works

- Write x as: $x = a_1 \mathbf{e}_1 + a_2 \mathbf{e}_2 + \ldots + a_n \mathbf{e}_n$
- Therefore, $\delta = Lx = \underbrace{\lambda_1 a_1 \mathbf{e_1} + \lambda_2 a_2 \mathbf{e_2}}_{\text{Small } \lambda_i \text{ – low frequencies}} + \ldots + \underbrace{\lambda_{n-1} a_{n-1} \mathbf{e_{n-1}} + \lambda_n a_n \mathbf{e_n}}_{\text{large } \lambda_i \text{ – high frequencies}}$
- Quantization error for δ $(\delta \to \delta + q_\delta)$ is:

$$q_\delta = \underbrace{c_1 \mathbf{e}_1 + c_2 \mathbf{e}_2}_{\text{low frequencies – small } c_i} + \ldots + \underbrace{c_{n-1} \mathbf{e}_{n-1} + c_n \mathbf{e}_n}_{\text{high frequency error – here } c_i \text{ are large}}$$

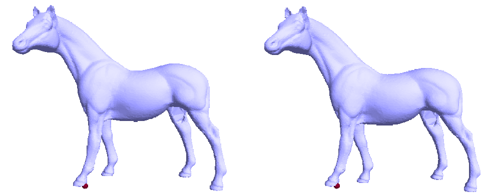
- Resulting error in x:

$$q_x = L^{-1} q_\delta = \underbrace{(1/\lambda_1) c_1 \mathbf{e}_1 + (1/\lambda_2) c_2 \mathbf{e}_2}_{(1/\lambda_i) \text{ is large – amplifies low-frequency errors}} + \ldots + \underbrace{(1/\lambda_{n-1}) c_{n-1} \mathbf{e}_{n-1} + (1/\lambda_n) c_n \mathbf{e}_n}_{(1/\lambda_i) \text{ is small – attenuates high-frequency errors}}$$

Thus, the error in x will contain **strong low**-frequency components but **weak high**-frequency components.

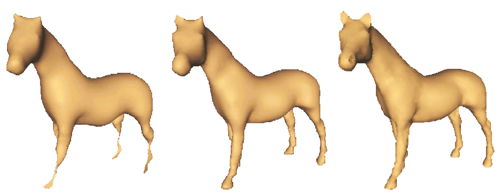
Spectral mesh compression

- Low-frequency errors are hard to see



Progressive transmission

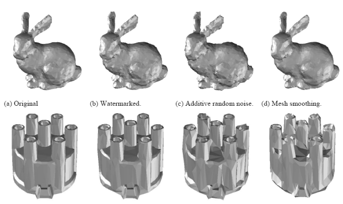
- First transmit the lower-eigenvalue coefficients (low frequency components), then gradually add finer details by transmitting more coefficients.



[Karni and Gotsman 00]

Mesh watermarking

- Embed a bitstring in the low-frequency coefficients
  - Low-frequency changes are hard to notice



(a) Original (b) Watermarked. (c) Additive random noise. (d) Mesh smoothing.

(e) Original (f) Watermarked. (g) Additive random noise. (h) Mesh smoothing.

[Ohbuchi et al. 2003]

Caveat

- Performing spectral decomposition of a large matrix ($n>1000$) is prohibitively expensive ($O(n^3)$)
  - Today's meshes come with 50,000 and more vertices
  - We don't want the decompressor to work forever!
- Possible solutions:
  - Simplify mesh
  - Work on small blocks (like JPEG)