Signal Processing for Meshes

Jason Lawrence
CS451: Advanced Computer Graphics
Spring 2008
Signal Processing

- Analysis, interpretation, manipulation of signals
  - sampling theory
  - aliasing / the Nyquist limit
  - filtering
  - ...

- Theoretical foundation: **Fourier Theory**
Refresher: Image Filtering

Original: Mandrill

Smoothed with Gaussian kernel
Refresher: Image Filtering

\[ f(x, y) \]

\[ g(x, y) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2 + y^2}{2\sigma^2}} \]

\[ f(x, y) \circ g(x, y) \]
**Convolution Theorem**

\[ f(x) \circ g(x) = \int_{t=-\infty}^{\infty} f(t) g(x-t) \, dt \]

\[ \mathcal{F}(f(x)) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{i\omega x} \, dx \]

\[ \mathcal{F}(f(x) \circ g(x)) = \mathcal{F}(f(x)) \cdot \mathcal{F}(g(x)) \]

\[ f(x) \circ g(x) = \mathcal{F}^{-1}(\mathcal{F}(f(x)) \cdot \mathcal{F}(g(x))) \]
Convolution Theorem

\[ \text{Amplitude} \times \text{Amplitude} = \text{Amplitude} \]
Regularly Sampled Data

Ideas extend to data arranged on any n-D regular grid:

\[ f(t) \]

\[ f(x, y) \]

\[ f(x, y, t) \]
What about 3D Geometry?

\[ f(s) = ?? \]
Mesh Smoothing
Basic Notation

\[ G = (V, E) \]
Basic Notation

\[ G = (V, E) \]
\[ i^* = \{ j : (i, j) \in E \} \]
Laplacian Operator

\[ G = (V, E) \]

\[ i^* = \{ j : (i, j) \in E \} \]

\[ \Delta x_i = \sum_{j \in i^*} w_{ij} (x_j - x_i) \]

\[ \sum_{j \in i^*} w_{ij} = 1 \]

“umbrella” operator
note that $x \rightarrow \Delta x$ operates on coordinates independently
Laplacian Operator

\[ K = I - W \]

\[ W_{ij} = w_{ij} \]

\[ \Delta x = -K x \]
Discrete Fourier Transform

\[ K = I - W \]

\( e^1, e^2, \ldots, e^n \)
eigenvectors of \( K \) form natural vibration modes

\( \lambda^1, \lambda^2, \ldots, \lambda^n \)
eigenvalues of \( K \) form natural frequencies
Smoothing with DFT

any graph signal can be represented in this basis smoothing = truncating eigenvectors w/ smaller eigenvalues!!!

DFT

Fourier matrix

\[ x = \sum_{j=1}^{n} \hat{x}_j e^{j} = \mathbf{E} \hat{x} \]
Smoothing with DFT

\[ K = I - W \]

\[ e^1, e^2, \ldots, e^n \]

\[ x = \sum_{j=1}^{n} \hat{x}_j e^j = E \hat{x} \]

is this efficient?

what is the bottleneck?
Smoothing with Diffusion

- Explicit Euler iterations:

\[ V_{t+dt} = V_t + \lambda L(V_t) dt \]

- Small time steps to meet stability requirements

\[ \lambda dt < 1 \]
Explicit Solver

- Explicit integration:

\[ V_{n+1} = V_n + \lambda d t L(V_n) \]

\[ V_{n+1} = (I + \lambda d t L)V_n \]

- What is involved in this, computationally?
- Drawback: small time steps => many iterations!
Results

Figures

10, 50, 200 diffusion steps
Implicit Solver

- Implicit integration:

\[ V_{n+1} = V_n + \lambda dtL(V_{n+1}) \]

\[ (I - \lambda dtL)V_{n+1} = V_n \]

- Now what is involved, computationally?
- No more timestep worries!
- The quantity \( \lambda dt \) controls “amount of smoothing”
Stability

“explicit” integration scheme

\[ y(t + dt) = y(t) + \dot{y}(t)dt \]

“implicit” integration scheme

\[ y(t + dt) = y(t) + \dot{y}(t + dt)dt \]

Slide Credit: Eitan Grinspun, Peter Schröder, and Mark Meyer
Signal Analysis

- Benefit of implicit scheme evident in transfer functions:

\[
\text{explicit} \quad 1 - \lambda dt \omega^2
\]

\[
\text{implicit} \quad \frac{1}{1 + \lambda dt \omega^2}
\]
Comparison

initial mesh

10 explicit iterations

1 implicit iteration
Results
Results
## Results

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Nb of faces</th>
<th>$\lambda dt = 10$</th>
<th>$\lambda dt = 100$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horse</td>
<td>42,000</td>
<td>8 iterations (2.86s)</td>
<td>37 iterations (12.6s)</td>
</tr>
<tr>
<td>Dragon</td>
<td>42,000</td>
<td>8 iterations (2.98s)</td>
<td>39 iterations (13.82s)</td>
</tr>
<tr>
<td>Isis</td>
<td>50,000</td>
<td>9 iterations (3.84s)</td>
<td>37 iterations (15.09s)</td>
</tr>
<tr>
<td>Bunny</td>
<td>66,000</td>
<td>7 iterations (4.53s)</td>
<td>35 iterations (21.34s)</td>
</tr>
<tr>
<td>Buddha</td>
<td>290,000</td>
<td>5 iterations (13.78s)</td>
<td>28 iterations (69.93s)</td>
</tr>
</tbody>
</table>
“Regular” Diffusion

- Problem with umbrella operator
  - does not distinguish these cases:

- in other words... assumes a regular surface parameterization
“Regular” Diffusion

initial mesh

umbrella smoothing

Desbrun
Length-Scale Weights

Define edge weights w.r.t. the edge length:

\[ w_{ij} = \frac{l(E_{ij})}{\sum_{j \in i^*} w_{ij}} \]

...but, now even tinier time steps:

\[ dt < \frac{l^2_{\text{min}}}{\lambda} \]
Scale-dependent Smoothing

initial mesh

umbrella smoothing with edge-length weights

Desbrun