Mesh Shape Editing

Computer Animation and Visualisation

Lecture 14
Taku Komura
Today

– Laplacian mesh editing
– As-rigid-as possible shape interpolation
– Deformation transfer
– Generalized Barycentric Coordinates
Editing shapes

• For animating rigid/articulated objects, we can use previous techniques like skinning
• Let’s think of animating cloths, clay, rubber, soft tissues, dolls, etc
Editing shapes

• Usually, it is easier to edit a given shape rather than modeling it from the beginning

• What is important when editing shapes?

• Keeping the local information unchanged
  – For faces, the relative location of the eyes, nose, and mouth must be similar to the original mesh
    • As-rigid-as possible
    • Laplacian coordinates
Surface Editing

• Must
  – Be fast (interactive rate)
  – Preserving the details

  -> Differential coordinates – Laplacian coordinates
  – Linear, invariant to scale, rotation [Sorkine ’04]
Laplacian coordinates

- Assuming all polygons are split into triangles
- Every vertex $v_i$ is surrounded by a group of vertices $\mathbb{N}_i$
- Coordinate $i$ will be represented by the difference between $v_i$ and the average of its neighbors

- The transformation between the Laplacian coordinates and the original vertices is linear

$$\mathcal{L}(v_i) = v_i - \frac{1}{d_i} \sum_{j \in \mathbb{N}_i} v_j.$$
Laplacian coordinates (2)

- Suppose the new position of the vertices are \( v'_i \)
- We want to keep the Laplacian coordinates the same after the deformation

\[
\delta_i = \mathcal{L}(v_i).
\]

\[
E(V') = \sum_{i=1}^{n} \left\| T_i(V') \delta_i - \mathcal{L}(v'_i) \right\|^2
\]

- \( T \) is a homogeneous transformation of scaling, rotation, and translation
Laplacian coordinates (3)

• We also want to constrain the position of some points (keeping $v_i'$ close to $u_i$)

• After all, we need to minimize the following function

\[
E(V') = \sum_{i=1}^{n} \| T_i(V') \delta_i - \mathcal{L}(v'_i) \|^2 + \sum_{i=m}^{n} \| v'_i - u_i \|^2.
\]

• This becomes a simple quadratic optimization problem
Solving a quadratic problem

Rewriting this problem, it becomes like

\[ \min_x \| c - A x \|^2 \]

The optimal \( x \) can be computed by solving

\[ A^T A x = A^T c \]
Editing Surfaces by Keeping the Details

- The user specifies the region of interest (ROI) (the area to be edited)
- The user directly moves some of the vertices
- The rest of ROI is decided by minimizing the error function

(a) (b) (c)
Some more ... 

http://www.youtube.com/watch?v=Yn3P4EK8sYE&feature=channel_page
It is possible to add more constraints

- Sometimes the shape might shrink as we allow scaling for the transformation
- It might be better to keep the volume the same
Creating a Volumetric Graph

- Add internal vertices and edges
- Compute and preserve the details for the internal structure

Figure 5: Volumetric graph construction.
Some Results

Figure 2: Large twist deformation.

Figure 3: Large bend deformation.
Today

- Laplacian mesh editing
- As-rigid-as possible shape interpolation
- Deformation Transfer
- Generalized Barycentric Coordinates
What if we want to interpolate different shapes?

– Not editing the shapes but need to morph it to the target shape
As-Rigid-As Possible Shape Interpolation [Alexa ’00]

• Interpolating the shapes of the two polygons so that each triangle (2D) / tetrahedron (3D) transformation appears as rigid as possible
Interpolating the shapes of triangles

Represent the interpolation of triangles by rotation & scaling

• linear interpolation

• rotation & scaling
**Background Knowledge:**

**2x2 matrix**

### Scaling

Scaling can be represented by a matrix $S$ as follows:

$$S = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

### Rotation

Rotation can be represented by a matrix $R_\alpha$ as follows:

$$R_\alpha = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$
Transformation by 2x2 Matrix

If the matrix is symmetric, can be decomposed as

\[ A = R \Lambda R^T, \quad R \text{ orthogonal}, \quad \text{where } \Lambda \text{ scaling matrix} \]

(true if \( A \) is symmetric)

The eigenvectors of \( A \) are the axes of the ellipse

\[ A = R \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} R^T \]
Geometric analysis of linear transformations

If the matrix is symmetric, it can be decomposed as

$$A = R \Lambda R^T, \quad R \text{ orthogonal}$$

(true if $A$ is symmetric)

The eigenvectors of $A$ are the axes of the ellipse.
General linear transformations: Singular Value Decomposition (SVD)

In general $A$ will also contain rotations, not just scales:

\[ A = U \sum V^T \]
General Transformation: SVD

\[ A = U \Sigma V^T \]
Back to ARAP Deformation
Least-Distorting Triangle-to-Triangle Morphing

• Say the three vertices \( \mathbf{P}=(p_1,p_2,p_3) \) are morphed to \( \mathbf{Q}=(q_1,q_2,q_3) \)

We want to compute a transformation that produces \( (q_2-q_1,q_3-q_1) = A (p_2-p_1, p_3-p_1) \)

• We can apply SVD to compute the scaling and rotation part.

\[
A = RS
\]

\[
A = U \Sigma V^T = UV^T \Sigma V^T = RS
\]
Interpolation of the transformation matrix

The intermediate vertices will be computed by

\[ V(t) = A(t) \cdot P \]

\( A \) is decomposed into the rotation part \( R(t) \) and scaling part and \( S \).

\[ A_{\gamma}(t) = R_{t\gamma}((1 - t)I + tS) \]
Keeping the transformation similar to $A_{i,j,k}$

- We cannot just interpolate all the triangles independently
- We need to move the vertices, not the triangles
- A vertex configuration that minimizes the error between $A$ and $B$ is computed

$$E_V(t) = \sum_{\{i,j,k\} \in T} \left\| A_{i,j,k}(t) - B_{i,j,k}(t) \right\|^2$$
Some 2D results
3D objects

- The method is applicable to polyhedra
- Tetrahedralization is applied to polyhedra and the tetrahedra are morphed so that they are as rigid as possible
More results
As-rigid-as possible shape manipulation [Igarashi et al. 05]

- interactive manipulation of characters based on the “as-rigid-as-possible” concept
  - [http://www-ui.is.s.u-tokyo.ac.jp/~takeo/research/rigid/index.html](http://www-ui.is.s.u-tokyo.ac.jp/~takeo/research/rigid/index.html)
  - [http://www.youtube.com/watch?v=1M_oyUEOHK8](http://www.youtube.com/watch?v=1M_oyUEOHK8)
Today

– Laplacian mesh editing
– As-rigid-as possible shape interpolation
– Deformation transfer
– Generalized Barycentric Coordinates
Deformation Transfer

A standard approach to apply the deformation of one object to another

Source deformed \rightarrow \text{Deformation Transfer} \rightarrow \text{Target deformed}
What does it do?

Given a source mesh in a reference pose and several deformations of it:

Given another mesh, called Target in same reference pose:

Reference

Source

Target

Roni Oeschger, Nov. 2005
What does it do?

Transfer the deformation to the target

Reuse the deformation that was created with probably a lot of effort
No need of Skeleton

Deformation Transfer is purely *mesh-based*

No need for an underlying skeleton structure
Not skeletal-driven deformations
– non-rigid or facial deformations etc
Approach

Compute the deformation $Q$ for every source triangle (orientation, scale, skew)

• Apply $Q$ to the corresponding target triangles
Approach

Deformation based on *per-triangle affine transformation*

\[ Qv_1 + d = \tilde{v}_1 \]

Source \[ \rightarrow \] Deformed Source
Deformation Gradient

“Deformation Gradient” $Q$ depends on
– triangle in reference pose
– triangle in deformed pose
Computing the Deformation Gradient

- Let $\mathbf{v}_i$ and $\mathbf{\tilde{v}}_i$, $i \in 1\ldots3$, be the undeformed and deformed vertices of the triangle.
- We compute a fourth undeformed vertex as
  \[
  \mathbf{v}_4 = \mathbf{v}_1 + (\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}_1) / \sqrt{|(\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}_1)|}
  \]
- The deformation gradient can be then computed by
  \[
  \mathbf{Q} = \mathbf{\tilde{V}} \mathbf{V}^{-1}
  \]
  \[
  \mathbf{V} = \begin{bmatrix} \mathbf{v}_2 - \mathbf{v}_1 & \mathbf{v}_3 - \mathbf{v}_1 & \mathbf{v}_4 - \mathbf{v}_1 \end{bmatrix}
  \]
  \[
  \mathbf{\tilde{V}} = \begin{bmatrix} \mathbf{\tilde{v}}_2 - \mathbf{\tilde{v}}_1 & \mathbf{\tilde{v}}_3 - \mathbf{\tilde{v}}_1 & \mathbf{\tilde{v}}_4 - \mathbf{\tilde{v}}_1 \end{bmatrix}
  \]
What happens if we apply Q to the target mesh?

Given a source mesh in a reference pose and several deformations of it:

Given another mesh, called Target in same reference pose:

Deformation Details

\[ Qv_1 + d = \nabla_1 \]

Source \[ \rightarrow \]
Deformed Source
Resulting Meshes

Leads to holes in the resulting mesh (B)
Because each triangle is transformed independently…
Consistency needed

• Preservation of consistency leads to a optimization problem:
Vertex Optimization

• Use the vertices as the variables
• Optimize the vertex locations such that the
  • deformation gradient of the source $S$ and
  • deformation gradient of the target $T$ are as similar as possible
• We solve the following quadratic problem:

$$\min_{\tilde{v}_1 \ldots \tilde{v}_n} \sum_{j=1}^{M} \|S_{s_j} - T_{t_j}\|_F^2.$$
Vertex Optimization (2)

Remember deformation gradient is computed by

\[ T = \tilde{V} V^{-1} \]
\[ V = [v_2 - v_1 \ v_3 - v_1 \ v_4 - v_1] \]
\[ \tilde{V} = [\tilde{v}_2 - \tilde{v}_1 \ \tilde{v}_3 - \tilde{v}_1 \ \tilde{v}_4 - \tilde{v}_1] \]

\( V \) is known, \( \tilde{V} \) is unknown for \( T \)

So \( T \) is linear with respect to \( \tilde{V} \)

Thus the following equation is a quadratic optimization problem with respect to \( \tilde{V} \)

\[
\min_{\tilde{v}_1 \ldots \tilde{v}_n} \sum_{j=1}^{\left| M \right|} \left\| S_{S,j} - T_{t,j} \right\|_F^2.
\]
Vertex Optimization (3)

Rewriting this problem, it becomes like

$$\min_{\tilde{v}_1 \ldots \tilde{v}_n} \| c - A\tilde{x} \|^2_2$$

where $\tilde{x}$ are the unknowns (components of $\tilde{v}$), the rest are known.

We can solve this kind of problem by solving a linear equation

$$A^T A\tilde{x} = A^T c$$
Deformation Transfer: Results

http://people.csail.mit.edu/sumner/research/deftransfer/
Today

– Laplacian mesh editing
– As-rigid-as possible shape interpolation
– Deformation Transfer
– Generalized Barycentric Coordinates
Barycentric Coordinates

Given a polygon with \( N \) vertices, the barycentric coordinates is a \( N \) dimensional vector.

We can use it to represent a position in the space with respect to the \( N \) points of the polygon.

We can also use it to interpolate attribute values defined at the \( N \) vertices of the polygon.
Interpolation

**Geometric interpolation** – derive the global coordinates for a position in parametric cell space

\[ v = \sum_i w_i p_i \]

**Attribute interpolation** – derive the attribute value for a position defined in parametric cell space

\[ f[v] = \sum_i w_i f_i \]

The \( w_i \) are called barycentric coordinates.

Demo [http://www.lidberg.se/math/shapetransforms/barycentric.html](http://www.lidberg.se/math/shapetransforms/barycentric.html)
Barycentric Coordinates

- **Requirements**
  - $W_i = 1$ & $W_j = 0$ when $p = p_i$ and $i \neq j$
  - When position is $p_i$ all other weights must be 0 so that $f = f_i$
  - Interpolated value is no less than minimum $f = f_{\min}$ and no greater than maximum $f = f_{\max}$
  - I.e. $f_{\min} \leq f_i \leq f_{\max}$

- **Weights bounded as follows:**

$$\sum W_i = 1, \ 0 \leq W_i \leq 1$$
## Applications

### Surface Deformation

<table>
<thead>
<tr>
<th>Control Mesh</th>
<th>Surface</th>
<th>Computing Weights</th>
<th>Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>216 triangles</td>
<td>30,000 triangles</td>
<td>1.9 seconds</td>
<td>0.03 seconds</td>
</tr>
</tbody>
</table>
## Applications

### Surface Deformation

<table>
<thead>
<tr>
<th>Control Mesh</th>
<th>Surface</th>
<th>Computing Weights</th>
<th>Deformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>98 triangles</td>
<td>96,966 triangles</td>
<td>3.3 seconds</td>
<td>0.09 seconds</td>
</tr>
</tbody>
</table>
Applications
Boundary Value Problems
Applications
Solid Textures
Interpolating the texture coordinates
Extend texture to interior
How to compute the barycentric coordinates for each point?
Wachspress coordinates

Only works with convex polygons

\[
\lambda_i(p) = \frac{A(v_{i-1}, v_i, v_{i+1})}{A(v_{i-1}, v_i, p)A(v_i, v_{i+1}, p)}
\]

\(A(a,b,c)\) is the signed area of triangle abc

\[
w_i(p) = \frac{\lambda_i(p)}{\sum_{j=1}^{n} \lambda_j(p)}
\]

http://www.lidberg.se/math/shapetransforms/wachspress.html
Mean Value Coordinates

A good and smooth barycentric coordinates that can smoothly interpolate the boundary values

Also works well for concave polygons

There is also a 3D version

\[ \lambda_i = \frac{w_i}{\sum_{j=1}^{k} w_j}, \quad w_i = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{||v_i - v_0||}, \]
Comparison

[Wachspress 1975]

Mean Value Coordinates
[Floater 2003, Hormann 2004]
Previous Work

[Wachspress 1975]

Mean Value Coordinates
[Floater 2003, Hormann 2004]
Previous Work

[Wachspress 1975]

Mean Value Coordinates

[Floater 2003, Hormann 2004]
Previous Work

[Wachspress 1975]

Mean Value Coordinates
[Floater 2003, Hormann 2004]
Application:
Surface Deformation

\[ v = \sum_{i} w_i p_i \]
Application:
Surface Deformation

\[ v = \sum_i w_i p_i \]
Application:
Surface Deformation

\[ v = \sum_i w_i p_i \]
Application:
Surface Deformation

\[ \mathbf{v} = \sum_i w_i \mathbf{p}_i \]
Interpolation: problem polygon

Problem: This vertex has too strong an influence in the red region

http://www.youtube.com/watch?v=egf4m6zVHUI

- Shows the problem of mean value coordinates at 1:54
- ill-formed polygon
  - poor interpolation results based on point distance due to narrow concavity
Harmonic Coordinates

The problem with Mean Value Coordinates is that their values are affected by the Euclidean distance but not the distance that needs to be travelled. Affected by geometrically close points.

Harmonic Scalar Field
- The value depends on the distance travelled inside the polygon.
Not affected by the Euclidean distance but the geodesic distance.
Harmonic Coordinates: Procedure

- For each vertex $i$ of the cage, set the potential value of $v_i$ to 1, and the rest to 0.
- Compute the potential for all the points inside the the polygon $(p_i)$ by solving a Laplace equation (as taught in Lecture 12).

$(p_1, p_2, \ldots, p_n)$ is the harmonic coordinates for each point after normalization.

The global position of the point can be expressed in the form

$$P = p_1 v_1 + p_2 v_2 + \ldots + p_n v_n$$
Harmonic Coordinates: comparison

Before editing  Edited by MVC  Edited by HC
Barycentric Coordinates

Summary

Triangulation / Tetrahedralization
   Only C0 continuous at the boundaries

Wachspress coordinates
   Can only handle convex objects

Mean value coordinates
   Can handle concave objects well to some extent
   Values affected by geometrically close control points
   Defined outside the polygon too

Harmonic Coordinates
   Can handle concave objects well
   Only defined inside the polygon
Summary

Modelling surfaces

• Deforming surfaces
  – Laplacian Coordinates
  – As-rigid-as possible shape interpolation

• Generalized Barycentric Coordinates
Readings

• Alexa et al. “As-rigid-as-possible shape interpolation SIGGRAPH ’00
• Deformation Transfer for Triangle Mesh, Sumner et al. SIGGRAPH 2004
• Igarashi et al. “As-rigid-as-possible shape manipulation”, SIGGRAPH ‘05
• Large Mesh Deformation Using the Volumetric Graph Laplacian
• Harmoinc Coordinates for Character Articulation: Joshi et al. SIGGRAPH 2007
• Mean Value Coordinates for Closed Triangular Meshes, Ju et al. SIGGRAPH 2005