12.2 Space Deformation
Hoooray!
Last Time

Surface Deformations
• Displacement function defined on the ambient space

\[ d : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \]

• Evaluate the function on the points of the shape embedded in the space

\[ \mathbf{x}' = \mathbf{x} + d(\mathbf{x}) \]

Twist warp
Global and local deformation of solids
[A. Barr, SIGGRAPH 84]
Freeform Deformation

- Control object
- User defines displacements $d_i$ for each element of the control object
- Displacements are interpolated to the entire space using basis functions $B_i(x) : \mathbb{R}^3 \rightarrow \mathbb{R}$

$$d(x) = \sum_{i=1}^{k} d_i B_i(x)$$

- Basis functions should be smooth for aesthetic results
Freeform Deformation

- Control object = lattice
- Basis functions $B_i(x)$ are trivariate tensor-product splines:

$$d(x, y, z) = \sum_{i=0}^{l} \sum_{j=0}^{m} \sum_{k=0}^{n} d_{ijk} N_i(x) N_j(y) N_k(z)$$

[Sederberg & Parry 86]
Freeform Deformation

- Aliasing artifacts
- Interpolate deformation constraints?
  - Only in least squares sense
Limitations of Lattices as Control Objects

- Difficult to manipulate
- The control object is not related to the shape of the edited object
- Parts of the shape in close Euclidean distance always deform similarly, even if geodesically far
● Control objects are arbitrary space curves
● Can place curves along meaningful features of the edited object
● Smooth deformations around the curve with decreasing influence
- Wish list for the displacement function $d(x)$:
  - Interpolate prescribed constraints
  - Smooth, intuitive deformation

$d(x_i) = d_i$
Volumetric Energy Minimization

- Minimize similar energies to surface case

\[ \int_{\mathbb{R}^3} \left( \|d_{xx}\|^2 + \|d_{xy}\|^2 + \ldots + \|d_{zz}\|^2 \right) \, dx \, dy \, dz \rightarrow \text{min} \]

- But displacements function lives in 3D…
  - Need a volumetric space tessellation?
  - No, same functionality provided by RBFs!

[RBF, Botsch & Kobbelt 05]
Radial Basis Functions

- Represent deformation by RBFs

\[ d(x) = \sum_{j} w_j \varphi(\|c_j - x\|) + p(x) \]

- Triharmonic basis function \( \varphi(r) = r^3 \)
  - \( C^2 \) boundary constraints
  - Highly smooth / fair interpolation

\[ \int_{\mathbb{R}^3} \left( \|d_{xxx}\|^2 + \|d_{xyy}\|^2 + \ldots + \|d_{zzz}\|^2 \right) \, dx \, dy \, dz \rightarrow \min \]
Radial Basis Functions

- Represent deformation by RBFs
  \[ d(x) = \sum_j w_j \varphi(\|c_j - x\|) + p(x) \]

- RBF fitting
  - Interpolate displacement constraints
  - Solve linear system for \( w_j \) and \( p \)

[RBF, Botsch & Kobbelt 05]
Radial Basis Functions

- Represent deformation by RBFs
  \[ d(x) = \sum_{j} w_j \varphi(\|c_j - x\|) + p(x) \]

- RBF evaluation
  - Function \( d \) transforms points
  - Jacobian \( \nabla d \) transforms normals
  - Precompute basis functions
  - Evaluate on the GPU!

[RBF, Botsch & Kobbelt 05]
Local & Global Deformations

[RBF, Botsch & Kobbelt 05]
Local & Global Deformations

[1M vertices movie]

[RBF, Botsch & Kobbelt 05]
Space Deformation

- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - ...

- Complexity mainly depends on the control object, not the surface

- 3M triangles
- 10k components
- Not oriented
- Not manifold
- Handle arbitrary input
  - Meshes (also non-manifold)
  - Point sets
  - Polygonal soups
  - ...

- Easier to analyze: functions on Euclidean domain
  - Volume preservation: $|\text{Jacobian}| = 1$

\[
F(x, y, z) = (F(x, y, z), G(x, y, z), H(x, y, z))
\]
then the Jacobian is the determinant

\[
\text{Jac}(F) = \begin{vmatrix}
\frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \\
\frac{\partial G}{\partial x} & \frac{\partial G}{\partial y} & \frac{\partial G}{\partial z} \\
\frac{\partial H}{\partial x} & \frac{\partial H}{\partial y} & \frac{\partial H}{\partial z}
\end{vmatrix}
\]
Space Deformation

- The deformation is only loosely aware of the shape that is being edited
- Small Euclidean distance $\rightarrow$ similar deformation
- Local surface detail may be distorted
Cage-Based Deformation

- Cage = crude version of the input shape
- Polytope (not a lattice)

[Ju et al. 05]
Each point $x$ in space is represented w.r.t. to the cage elements using coordinate functions.

$$x = \sum_{i=1}^{k} w_i(x) p_i$$
● Each point $\mathbf{x}$ in space is represented w.r.t. to the cage elements using coordinate functions

$$\mathbf{x} = \sum_{i=1}^{k} \omega_i(\mathbf{x}) \mathbf{p}_i$$
Cage-Based Deformation

\[ x' = \sum_{i=1}^{k} w_i(x) p'_i \]

[Ju et al. 05]
Generalized Barycentric Coordinates

- Lagrange property: $\omega_i(p_j) = \delta_{ij}$

- Reproduction: $\forall x, \sum_{i=1}^{k} \omega_i(x) p_i = x$

- Partition of unity: $\forall x, \sum_{i=1}^{k} \omega_i(x) = 1$
Mean-value coordinates
[Floater 2003, Ju et al. 2005]
- Generalization of barycentric coordinates
- Closed-form solution for $w_i(x)$
Coordinate Functions

- **Mean-value coordinates**
  [Floater, Ju et al. 2005]
  - Not necessarily positive on non-convex domains
Harmonic coordinates (Joshi et al. 2007)
- Harmonic functions $h_i(x)$ for each cage vertex $p_i$
- Solve $\Delta h = 0$
subject to: $h_i$ linear on the boundary s.t. $h_i(p_j) = \delta_{ij}$
Harmonic coordinates ([Joshi et al. 2007](#))

- Harmonic functions $h_i(x)$ for each cage vertex $p_i$
- Solve $\Delta h = 0$ subject to: $h_i$ linear on the boundary s.t. $h_i(p_j) = \delta_{ij}$

Volumetric Laplace equation

Discretization, no closed-form
Coordinate Functions

- Harmonic coordinates *(Joshi et al. 2007)*
Green coordinates ([Lipman et al. 2008](#))
Observation: previous vertex-based basis functions always lead to affine-invariance!

\[ x' = \sum_{i=1}^{k} w_i(x) p'_i. \]
- Green coordinates (Lipman et al. 2008)
- Correction: Make the coordinates depend on the cage faces as well

\[ x' = \sum_{i=1}^{k} w_i(x) p'_i + \sum_{j=1}^{m} \psi_j(x) n'_j \]
Coordinate Functions

- Green coordinates ([Lipman et al. 2008](#))
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D
Coordinate Functions

- Green coordinates (Lipman et al. 2008)
- Closed-form solution
- Conformal in 2D, quasi-conformal in 3D

Alternative interpretation in 2D via holomorphic functions and extension to point handles: Weber et al. Eurographics 2009
Cage-Based Methods: Summary

Pros:
- Nice control over volume
  - Squish/stretch

Cons:
- Hard to control details of embedded surface
Non-Linear Space Deformation

- Involve nonlinear optimization
- Enjoy the advantages of space warps
- Additionally, have shape-preserving properties
As-Rigid-As-Possible Deformation

Points or segments as control objects
First developed in 2D and later extended to 3D by Zhu and Gortler (2007)

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Attach an affine transformation to each point \( \mathbf{x} \in \mathbb{R}^3 \):

\[
A_x(p) = M_x p + t_x
\]

- The space warp:

\[
\mathbf{x} \rightarrow A_x(\mathbf{x})
\]
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Handles $p_i$ are displaced to $q_i$
- The local transformation at $x$:
  \[ A_x(p) = M_x p + t_x \text{ s.t.} \]
  \[
  \sum_{i=1}^{k} w_i(x) \left\| A_x(p_i) - q_i \right\|^2 \rightarrow \min
  \]
- The weights depend on $x$:
  \[ w_i(x) = \| p_i - x \|^{-2\alpha} \]
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- No additional restriction on $A_x(\cdot)$ – affine local transformations
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to similarity
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to similarity

$$M_x = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$
Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

- Restrict $A_x(\cdot)$ to rigid
As-Rigid-As-Possible Deformation

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]

Restrict $A_x(\cdot)$ to rigid

$$M_x = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$$

Solve for $M_x$ like similarity and then normalize
As-Rigid-As-Possible Deformation

- Examples

Moving-Least-Squares (MLS) approach [Schaefer et al. 2006]
No linear expression for similarity in 3D
Instead, can solve for the minimizing rotation by polar decomposition of the 3×3 covariance matrix

\[
\arg\min_{R \in \text{SO}(3)} \sum_{i=1}^{k} w_i(x) \|Rp_i - q_i\|^2
\]
Zhu and Gortler also replace the Euclidean distance in the weights by “distance within the shape”

\[ w_i(x) = d(p_i, x)^{-2\alpha} \]
Moving-Least-Squares (MLS) extension to 3D [Zhu & Gortler 07]

- More results
As-Rigid-As-Possible Deformation

- Surface handles as interface
- Underlying graph to represent the deformation; nodes store rigid transformations
- Decoupling of handles from def. representation

Deformation Graph

Optimization Procedure
Deformation Graph

Embedded Deformation [Sumner et al. 07]
Deformation Graph

Embedded Deformation [Sumner et al. 07]

Begin with an embedded object.
Deformation Graph

Embedded Deformation [Sumner et al. 07]

Begin with an embedded object.

Nodes selected via uniform sampling; located at $g_j$.

One rigid transformation for each node: $R_j, t_j$.

Each node deforms nearby space.

Edges connect nodes of overlapping influence.
Deformation Graph

Embedded Deformation [Sumner et al. 07]

Begin with an embedded object.

Nodes selected via uniform sampling; located at $g_j$.

One rigid transformation for each node: $R_j$, $t_j$.

Each node deforms nearby space.

Edges connect nodes of overlapping influence.
Deformation Graph

Influence of nearby transformations is blended.

\[ x' = \sum_{j=1}^{m} w_j(x) \left[ R_j(x - g_j) + g_j + t_j \right] \]

\[ w_j(x) = (1 - \|x - g_j\| / d_{\text{max}})^2 \]
Optimization

Embedded Deformation [Sumner et al. 07]

Select & drag vertices of embedded object.
Optimization

Embedded Deformation [Sumner et al. 07]

Select & drag vertices of embedded object.

Optimization finds deformation parameters $R_j$, $t_j$. 
Optimization

\[
\min_{R_1, t_1, \ldots, R_m, t_m} \ w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}
\]

Graph parameters  Rotation term  Regularization term  Constraint term

Select & drag vertices of embedded object.

Optimization finds deformation parameters \( R_j, t_j \).
Optimization

\[
\min_{R_1, t_1, \ldots, R_m, t_m} \sum_{j=1}^{m} \mathcal{E}\text{rot} + \sum_{j=1}^{m} \mathcal{E}\text{reg} + \sum_{j=1}^{m} \mathcal{E}\text{con}
\]

\[
\text{Rot}(\mathbf{R}) = (c_1 \cdot c_2)^2 + (c_1 \cdot c_3)^2 + (c_2 \cdot c_3)^2 + (c_1 \cdot c_1 - 1)^2 + (c_2 \cdot c_2 - 1)^2 + (c_3 \cdot c_3 - 1)^2
\]

\[
\mathcal{E}\text{rot} = \sum_{j=1}^{m} \text{Rot}(\mathbf{R}_j)
\]

For detail preservation, features should rotate and not scale or skew.
Optimization

\[
\min_{R_1, t_1, \ldots, R_m, t_m} w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}
\]

\[
E_{\text{reg}} = \sum_{j=1}^{m} \sum_{k \in N(j)} \alpha_{jk} \left\| R_j (g_k - g_j) + g_j + t_j - (g_k + t_k) \right\|_2^2
\]

where node \( j \) thinks node \( k \) should go

where node \( k \) actually goes

Neighboring nodes should agree on where they transform each other.

Embedded Deformation [Sumner et al. 07]
Optimization

\[
\min_{R_1, t_1, \ldots, R_m, t_m} \ w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}
\]

\[
E_{\text{con}} = \sum_{l=1}^{p} \left\| \tilde{\mathbf{v}}_{\text{index}(l)} - \mathbf{q}_l \right\|_2^2
\]

Handle vertices should go where the user puts them.
Optimization

\[
\min_{R_1, t_1, \ldots, R_m, t_m} \ w_{\text{rot}} E_{\text{rot}} + w_{\text{reg}} E_{\text{reg}} + w_{\text{con}} E_{\text{con}}
\]

Embedded Deformation [Sumner et al. 07]
Results on Polygon Soups

Embedded Deformation [Sumner et al. 07]
Results on Giant Mesh

Embedded Deformation [Sumner et al. 07]
Detail Preservation

Embedded Deformation [Sumner et al. 07]
• Decoupling of deformation complexity and model complexity

• Nonlinear energy optimization – results comparable to surface-based approaches
Projects
Goal

• Small research project

• 1 week for project proposal, **deadline April 3**
  • **choose between 3 options: A, B, or C**

• >1 month for project, **deadline May 08**

• group, size up to 2

• contributes **30%** to the final grade.

• send to tianyeli@usc.edu
A) For the disciplined

• Deformation Project, we will provide a framework
• You will implement a surface-based linear deformation algorithm (bending minimizing deformation).

B) For the creative [+10 points]

• Imagine an interesting topic around geometry processing or related to your PhD research or something you always wanted to do, and write a proposal.
• If it gets approved, you are good to go.

C) For the bad ass [+10 points]

• Implement a Siggraph, SGP, SCA, or Eurographics Paper.
• Geometry processing related of course ;-}
Project Submission

Deliverables for A)

- Source Code, Binary, Data
- Text files describing the project, how to run it.

Deliverables for B) and C)

- Short Presentation will be held May 8th (length TBD)
- Video / Figures
- Documentation (pdf, doc, txt file): 2 or more pages, short paper style, be rigorous and organized, must include at least abstract, methodology, and results.
Project Proposal

Structure

• Title
• Motivation
• Goal
• Proposed Method
• References

Format

• authors’ names/student IDs
• 1-2 pages
• .doc, .pdf, .txt
• figures
Deformation Framework for A)

- Inherit from MeshViewer with user interface:
  - ‘p’: pick a handle
  - ‘d’: drag a handle (last one with starting code)
  - ‘m’: move the mesh
Deformation Framework for A)

- add handle picking code to
  DeformationViewer::mouse()
- add deformation codes to
  DeformationViewer::deform_mesh()
- add extra classes and files if needed
- **gmm** is provided to solve linear systems
Some ideas for B) or C):

- **registration**: articulated / deformable motions…
- **shape matching**: RANSAC, spin images, spherical harmonics…
- **Smoothing**: implicit surface fairing…
- **parameterization**: harmonic/conformal mapping…
- **remeshing**: anisotropic, quad mesh…
- **deformation**: As-rigid-as-possible, gradient-based…
- …
http://cs621.hao-li.com

Thanks!