Example-based Dynamic Modeling

Zhang wenjing

School of Computer Engineering

Supervisor:
Prof. Zheng Jianmin

Co-Supervisor:
Prof. Nadia magnenat Thalmann

2011-10-18
Outline

• **Background & Related work**
  – Purely geometrical modeling
  – Physically based modeling

• **Our method**

• **Future work**
Our goal

- user desired deformation
- physical realism
- efficient
- dynamic

Good deformation
Background

- Traditional method use general numerical criteria

Laplace mesh editing [Sorkine, 2004]

As rigid as possible deformation [Sorkine, 2007]

Poisson based modeling [Yu, 2004]

Preserve the Laplacian coordinate
Background

• **Example-based deformation**

  - **Mesh-based inverse kinematic**
    - [summer, 2005]
    - Search the optimal deformation in the example space

  - **Example-based facial rigging**
    - [Li, 2010]
    - Use examples as training data

  - Intuitive and efficient

  ![Example-based deformation](image)
Background

All these are purely geometrical

No physical realism

Inertial

acceleration

velocity
Physically based modeling

- Use physical mechanics law to guide the deformation
  - Realistic
  - Natural choice for dynamic modeling
Physically based modeling

• Equations of motion \([\text{Euler, Lagrange}]\)

\[
M\ddot{x}(t) + R(x(t)) = f(t)
\]

• \(x\) = position of vertices
Problems with physically based modeling

- Hard to control the deformation
- Time consuming
Time consuming

\[ M\ddot{x}(t) + R(x(t)) = f(t) \]

- **High-dimensional of the ODEs**
  - \( x \) is of dimension \( 3n \) (\( n \) is number of vertices)

- **Not real-time for large models**
  - around 1K DOFs at most for real-time
Hard to control the deformation

- Material parameters
- Tedious to tune
- User desired deformation
  - Non-intuitive relationship
  - Numerical problem
  - Contradiction with physical law
- Deformation result
Combination

Example based modeling + Physically based modeling = Good deformation
Related work

• Example-based Elastic Material [Sebastian Martin, 2011]
  1. At every time step, projecting the pose onto the example space
  2. use the projection as a rest post to generated additional force
  3. Add the additional force as external force
• First try to combine example based and physically based modeling
• Drawback
  --time consuming

Our method

• Build an example space
• Build the dynamic equation in the example space

\[ \dot{\omega} = \omega \]

\[ M_G \dot{\omega} + R_{\text{int}}(\omega) = R_{\text{ext}} \]

\[ x = x(\omega) \]
Our method

• Given a based tetrahedron mesh $x_0$ and $l$ example poses $x_1, x_2, \ldots, x_l$. 
Our Method

- We first convert the example pose into feature vector:

\[ \hat{x}_i = \tau(x_i) \]

- A lot of kinds of feature vector can be used

- We use the feature vector proposed in MeshIk
  - the polar decomposition of the deformation gradient
  - It well preserve the meaningful information in mesh
  - The computation of its derivative is easy
Our Method

• Then we build an example space by linearly interpolating these feature vectors:

\[
\{x, w\} : x(w) = \tau^{-1}(\sum_i w_i \hat{x}_i)
\]

• \(w\) is the interpolation weight, which acts as coordinates in example space
Our method

- Build the motion equation using the interpolation weight:

\[ M_G \ddot{w} + R_{\text{int}}(w) = R_{\text{ext}} \]

- \( M_G \) ~ generalized mass matrix

- \( R_{\text{int}} \) ~ generalized internal force

- \( R_{\text{ext}} \) ~ generalized external force

- ODE dimension is the number of examples (typically small, around 10)
Our method

• For some application, extrapolation is poor:

• Add constraint to the generalized coordinate.

• E.g. one typical constraint is:

\[
\begin{cases}
  w_i \leq 1 \\
  w_i \geq 0 \\
  \sum_i w_i \leq 1
\end{cases}
\]
Time complexity

\[ M_G \ddot{\omega} + R_{\text{int}}(\omega) = R_{\text{ext}} \]

- \( M_G \sim \) constant
- \( R_{\text{int}} \sim O(l^3) \)
- \( R_{\text{ext}} \sim O(nl) \)
Fairness evaluation

**Good Deformation**

- User desired deformation
- Physical realism
- Dynamic
- Efficiency
- Interpolation of examples
- based on continuum mechanics
- Motion equation
- Low dimension of the ODE
### Preliminary result

- **CPU time**

<table>
<thead>
<tr>
<th>~3K vertex</th>
<th>~6K tetrahedron</th>
<th>~20fps</th>
<th>4 examples</th>
</tr>
</thead>
</table>

Demo 1  Demo 2
Future work

- Collision detection and response
  - easy for original system on vertices
  - however, nontrivial for generalized coordinates
    -- especially for nonlinear interpolation

- Extension to surface mesh
  - Designer work on surface mesh more often
  - Conversion from surface mesh to tetrahedron mesh is hard

- GPU implementation
Thank you !