A simple approach to mesh deformation

Shingo Ito

Table of contents

Introduction

- Surface deformation is an important research topic in shape and geometric modeling.
- Here we present a simple approach for mesh deformation based on surface reconstruction of a deformed sampled point-cloud.

Related works

- Mesh deformation through variational principles (as solution of the harmonic, biharmonic or polyharmonic equation),
- Cage-based deformation (using generalized barycentric coordinates),
- Computing the distance to the surface and deforming this distance field (level-set methods).

Algorithm overview

Algorithm 1 Overview

Sample from the surface

Apply the deformation field to the samples

Apply a surface reconstruction algorithm to the samples

Surface sampling

Algorithm 2 Uniform sampling from a triangle

function SAMPLE(p_1, p_2, p_3, ξ_1, ξ_2)

▷ p_1, p_2 and p_3 are the triangle vertices, ξ_1 and ξ_2 are samples from the unit uniform distribution

Compute the barycentric coordinates: $u = 1 - \sqrt{\xi_1}$ and $v = \xi_2\sqrt{\xi_1}$

Compute the sample: $P = u * p_1 + v * p_2 + (1 - u - v) * p_3$

return P

end function

Points deformation

Given a selected vertex x_S from the input surface, apply to each point a truncated Gaussian centered on this selected vertex in the normal direction:

$$f(x) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathbf{x}-\mathbf{x}_S)^2}{2\sigma^2}\right) & \text{if } \|\mathbf{x} - \mathbf{x}_S\| < r \\ 0 & \text{otherwise} \end{cases}$$

where r is the radius of influence for the truncated Gaussian.

Surface reconstruction

Implementation and experiments with three surface reconstruction algorithms.
All approaches are implicit surface based surface reconstruction algorithms. I.e. the output is a function f:

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}$$

The zero level-set of f corresponds to the surface of interest.

Hermite Radial Basis Functions (Macedo et al.)

Given a set of points $P = \mathbf{x}_i$ with normal vector at each point $N = \mathbf{n}_i$, look for a function: $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ defined as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i \phi(\|\mathbf{x} - \mathbf{x}_i\|) - \boldsymbol{\beta}_i \nabla \phi(\|\mathbf{x} - \mathbf{x}_i\|))$$

where the unknowns to be determined are the coefficients α_i and $\boldsymbol{\beta}_i$.

Hermite Radial Basis Functions

The conditions used to determine the coefficients are:

$$f(\mathbf{x}_i) = 0, \mathbf{x}_i \in \mathbb{R}^3$$

for interpolating points on the surface. And:

$$\nabla f(\mathbf{x}_i) = \mathbf{n}_i, \mathbf{x}_i \in \mathbb{R}^3, \mathbf{n}_i \in \mathbb{R}^3$$

for interpolating the normals on the surface.

Hermite Radial Basis Functions

This gives the following system of equations to solve:

$$f(\mathbf{x}_i) = \sum_{j=1}^{n} (\alpha_j \phi(\|\mathbf{x}_i - \mathbf{x}_j\|) - \boldsymbol{\beta}_j \nabla\phi(\|\mathbf{x}_i - \mathbf{x}_j\|)) = 0$$

$$\nabla f(\mathbf{x}_i) = \sum_{j=1}^{n} (\alpha_j \nabla\phi(\|\mathbf{x}_i - \mathbf{x}_j\|) - H\phi(\|\mathbf{x}_i - \mathbf{x}_j\|)\boldsymbol{\beta}_j) = \mathbf{n}_i$$

where H is the Hessian matrix of ϕ.

Hermite Radial Basis Functions

Adding linear polynomials and the additional condition:

$$\sum_{j=1}^{n} [p_k(x_j) \nabla p_k(x_j)^T] \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} = 0 \tag{1}$$

We obtain:

$$\sum_{j=1}^{n} \begin{bmatrix} \phi(\|\mathbf{x}_i - \mathbf{x}_j\|) & -\nabla\phi(\|\mathbf{x}_i - \mathbf{x}_j\|)^T \\ \nabla\phi(\|\mathbf{x}_i - \mathbf{x}_j\|) & -H\phi(\|\mathbf{x}_i - \mathbf{x}_j\|) \end{bmatrix} \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix}$$
$$+ \sum_{l=1}^{m} \lambda_l \begin{bmatrix} p_l(\mathbf{x}_i) \\ \nabla p_l(\mathbf{x}_i) \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{n}_i \end{bmatrix} \tag{2}$$

Hermite Radial Basis Functions

One possible choice of basis function is the triharmonic splines:

$$\phi(\|\mathbf{x}\|) = \|\mathbf{x}\|^3$$

$$\nabla\phi(\|\mathbf{x}\|) = 3\mathbf{x}\|\mathbf{x}\|$$

$$H\phi(\|\mathbf{x}\|) = \begin{cases} 3/\|\mathbf{x}\|(\|\mathbf{x}\|^2 I_{3\times 3}) + \mathbf{x}\mathbf{x}^T, & \|\mathbf{x}\| \neq 0 \\ 0_{3\times 3}, & \|\mathbf{x}\| = 0 \end{cases}$$

Closed form approximation for compactly supported functions (Liu and Wang)

When compactly supported functions are used, it is possible to compute an approximation to the solution without forming or inverting any matrix

$$f(x) = -\sum_{j=1}^{n} < \frac{\rho_j^2}{20 + \eta\rho_j^2}\mathbf{n}_j, \nabla\phi_{\rho_j}(\|\mathbf{x} - \mathbf{x}_j\|) > \tag{3}$$

where ρ_j is the radius of support of the basis function associated to the center j, and ϕ is the compactly supported basis function.

$$\phi_\rho(r) = \phi(r/\rho)$$

where:

$$\phi(t) = \begin{cases} (1-t)^4(4t+1), & t \in [0,1] \\ 0, & \text{otherwise} \end{cases}$$

Poisson surface reconstruction (Kazhdan, Bolitho and Hoppe)

The third approach consists in computing an indicator function for the solid to be reconstructed:

$$f_S(\mathbf{x}) = \begin{cases} 1 & \mathbf{x} \in S \\ 0 & otherwise \end{cases}$$

by solving the Poisson equation

$$\Delta f = div(\mathbf{n})$$

where $\mathbf{n}$ is an extrapolation of the given normal vector field.

Meshing

To recover a triangle mesh for the deformed point-cloud and the corresponding function f, its zero level-set needs to be meshed.

Two approaches were used:

- Marching Cubes
- Delaunay based approach

Marching Cubes (Lorensen and Cline)

Algorithm for rendering isosurfaces from volumetric data.

- Compute a bounding box for the object to be meshed and subdivide it regularly into smaller cells.
- Sample the function f at the eight corners of each cell.
- If one or more values is less than the user-specified iso-value, and one or more have values is greater than this isovalue, the cell must intersect the isosurface.
- Determine the edges in the cell that are intersected by the isosurface.
- Connect the patches from all cells, we get a linear approximation of the isosurface.

Delaunay based implicit surface meshing

Marching Cubes based algorithm sample the function at each cells node. But it does not exploit the information about the surface location. Instead it seems preferrable to use the following approach:

- Compute a Delaunay tetrahedralization of the deformed point-cloud.
- Peel off outside tetrahedra using the fitted function

The environment used for prototype and in the experiments

The development and all experiments were run on a regular note PC.

CPU	Intel Corei5 1.4GHz
GPU	Intel HD Graphics5000
Memory	4.0GB RAM
OS	OS X Yosemite version 10.10.5
Programming Language	C++
Libraries	CGAL4.7, OpenGL, Eigen

Deformation of a sphere

Start from a sphere represented by a triangle mesh, made of 174 vertices and 344 triangles.

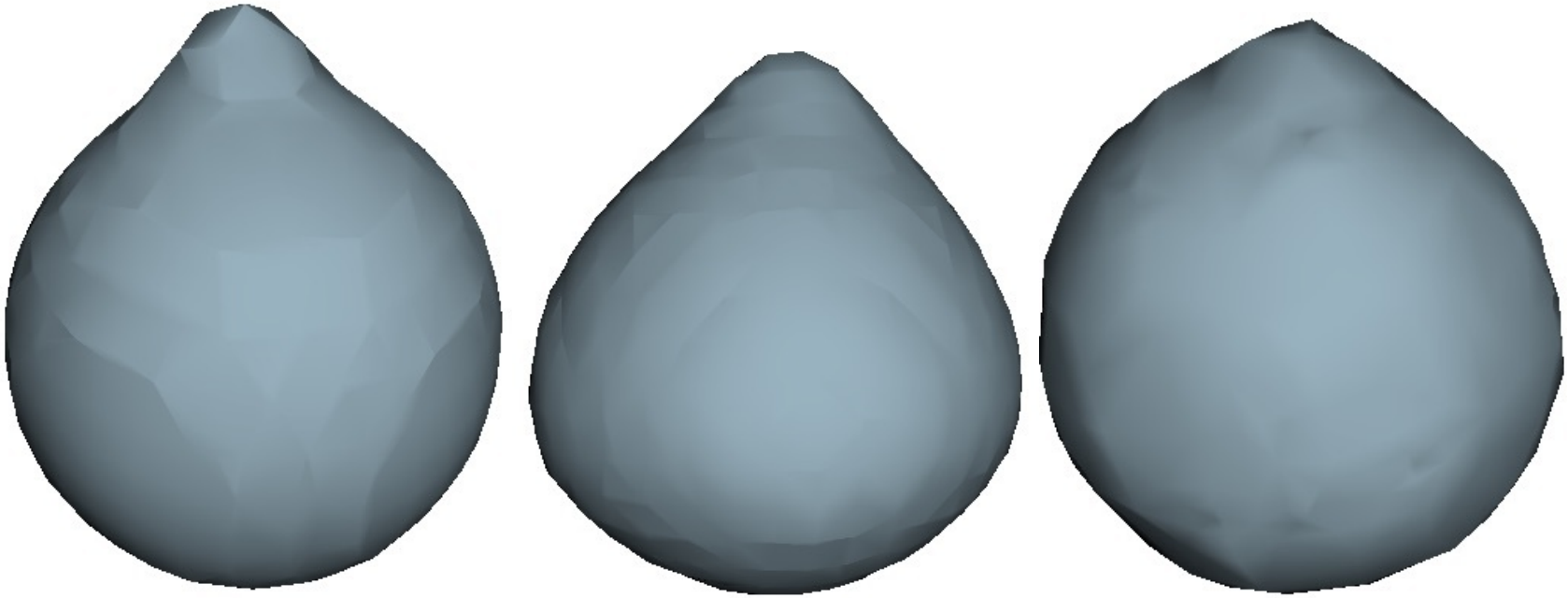

Figure: Deformed sphere using (from left to right): the HRBF approach, the closed form solution to the HRBF approach and the Poisson surface reconstruction approach

Example of sculpture

Figure: Example to model a simple character face

- The eyes and the mouth were carved by changing the sign of the field used for the deformation.
- Changing the width of the Gaussian function used for the deformation could allow to add or carve smaller details.

Deformation of a more complex surface

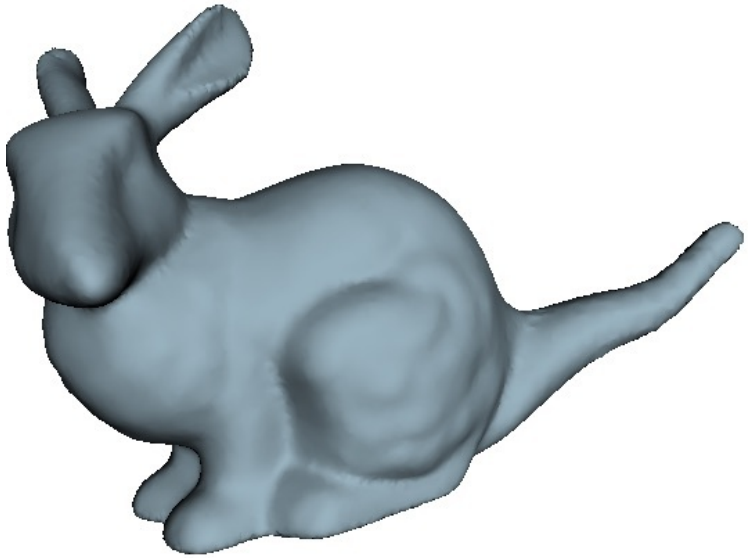

Figure: Deformed bunny object with stretched tail and nose

- The input mesh for this object contains 14072 vertices and 28042 triangles.
- Both the tail and the nose are stretched.

Conclusion

- A simple algorithm for mesh deformation.
- Sample from the mesh, deforms the sample (no connectivity to maintain), perform surface reconstruction.
- Future works : algorithmitic improvements, implementation of parts of the algorithm on the graphics card (GPU).

Thank you.