Conformal quaternion based mesh (re)construction

Adam Sturge[1]

[1]Faculty of Computer Science
University of Toronto

April 5 2017

About me

- My name is Adam Sturge
- I'm a MSc student in the numerical computing group (more statistics honestly)
- I majored in Applied mathematics, Physics, and Computer Science at the Memorial University of Newfoundland; my home province.
- I worked at EA games for about 3 years doing tools/automation/data analysis

What are quaternions?

- Quaternions are a Division Algebra
- You can consider them as an extension of the complex numbers with 3 complex coordinates instead of just 1
- $\lambda = a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k} = (a, b, c, d)$
- Multiplication is not commutative. $ab \neq ba$
- $\bar{\lambda} = a - b\boldsymbol{i} - c\boldsymbol{j} - d\boldsymbol{k}$
- $||\lambda|| = \sqrt{a^2 + b^2 + c^2 + d^2}$
- $\lambda^{-1} = \frac{\bar{\lambda}}{||\lambda||}$
- $\lambda \in \mathbb{H}$

Why are they useful?

- You can embed $\mathbb{R}^3$ in $Im\mathbb{H}$.
- Unit quaternions represent a rotation about any axis in $\mathbb{R}^3$. If $\vec{u}$ is the axis you want to rotate a vector $\vec{p}$ around by an angle θ then $\hat{p} = \lambda p \bar{\lambda}$, where $\lambda = (cos(\frac{\theta}{2}), u_x sin(\frac{\theta}{2}), u_y sin(\frac{\theta}{2}), u_z sin(\frac{\theta}{2}))$
- Non-unit quaternions represent a rotation and a scale when used in this way
- So this operation with quaternions represents a conformal map when applied to the vertices of a triangle

Spin Transformations of Discrete Surfaces

- Spin Transformations of Discrete Surfaces is a paper where the authors make use of this quaternionic representation of vectors.
- They build on previous results that show that 2 surfaces f and $\tilde{f}$ are conformally equivalent if their differentials df and $d\tilde{f}$ are related by

$$df = \lambda df \bar{\lambda} \tag{1}$$

- λ must apply globally to the whole surface, and as such equation 1 may not have a solution

Spin Transformations of Discrete Surfaces

- The authors introduce a linear integrability condition $(D - \rho)\lambda = 0$ that characterizes all valid quaternions λ as presribed changes in **mean curvature half density** ρ and the dirac operator D.
- Mean curvature half density: $H|df|$, where H is mean curvature. Is nicer to work with in this contest because a change in mean curvature alone could refer to deformation or just scaling. However a change in $H|df|$ is always indicative of deformation

Spin Transformations of Discrete Surfaces

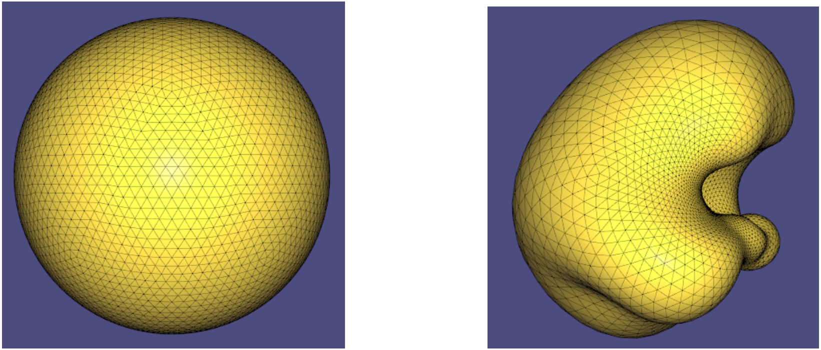

Figure: sphere deformation

My goal

- To use their deformation technique to continuously deform a given canonical topological object into a target mesh given a sampling of points from the target mesh
- It is often easy for human being to tell what topology a mesh should be from looking a point cloud, it's less obvious what geometry it should have.

Results

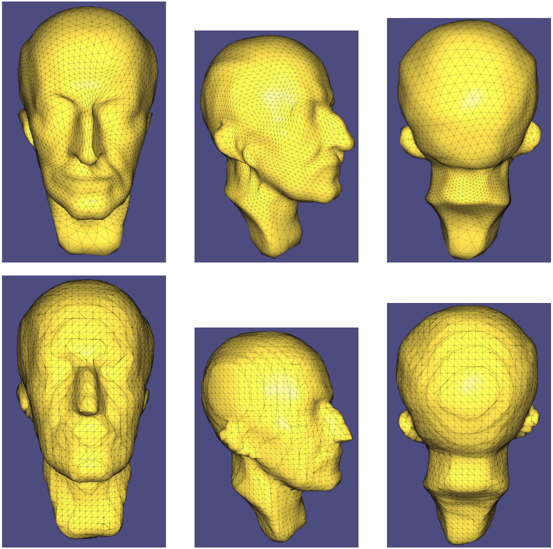

Figure: Top : quaternion deformation. Bottom : Poisson reconstruction assignment

Results

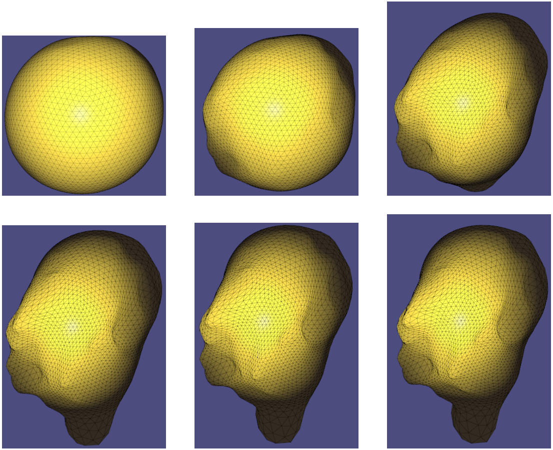

Figure: gradient decent iterations (with resampling): 10, 20, 30, 40, 50, 60

Results

DEMO

Mesh fitting procedure

- I decided to go with an Iterative Closest Point (ICP) algorithm to deform the meshes. There are two approaches
- First sample the **seed mesh** and for each sample find the closest point in the **target point cloud**
- First sample the **target point cloud** and project the samples onto **seed mesh**
- After collecting the samples, use them to minimize some energy

Minimization energy

- Energy to minimize (point to point)

$$E(\lambda) = \int_{\Omega} ||X(\lambda) - P||^2 d\Omega \approx \sum_{i=0}^{k} ||x_i(\lambda) - p_i||^2 \qquad (2)$$

- Energy to minimize (point to plane)

$$E(\lambda) = \sum_{i=0}^{k} ((x_i(\lambda) - p_i) \cdot n_i)^2 \qquad (3)$$

Discretization

- How to discretize this problem for triangle meshes?
- Assign each vertex v_i a quaternion λ_i
- Solve least squares problem for new vertex positions $\tilde{v}_i$
- Instead focus on finding a quaternion vector $\vec{\lambda} = (\lambda_0, \lambda_1, \ldots, \lambda_n)$ to deform vertices
- More details in Spin Transformations of Discrete Surfaces Keenan et al

How to minimize?

- The relationship between $\vec{\lambda}$ and $x_i(\vec{\lambda})$ is pretty complicated.
- $E(\vec{\lambda})$ is a functional, so we can use gradient decent
- $\vec{\lambda} \to \vec{\lambda} + \alpha \frac{\nabla E(\vec{\lambda})}{||\nabla E(\vec{\lambda})||}$
- Use reverse mode autodifferentiation to algorithmically compute gradient
- Start from identity transform where each $\lambda_i = (1, 0, 0, 0)$
- Do not need the linear integrability condition from Keenan et al since we are taking a guess and refining it

Issues

- Each iteration of gradient decent involves solving a large least squares problem, which can take up to 1 sec to solve
- Sampling can take a long time
- Storing the large matricies involved requires a lot of RAM, peaking around 20GB on my home PC. This could probably be mitigated somewhat be optimized code
- Gradient decent is prone to getting stuck in local optima (lack of fine detail)
- Introduces cloth like ripples onto the resulting surface

Future work

- Look into spatial data structures to speed up sampling
- Find a closed form solution for the gradient so I don't need autodifferentiion
- Find closed form equation for optimal $\vec{\lambda}$ instead of using gradient decent