[CSCI 2240] Assignment 3: As-Rigid-As-Possible Shape Manipulation (rigid)

Released: 3/8/19
Due: 3/15/19

In this assignment, you will implement the algorithm described in the paper "As-Rigid-As-Possible Shape Manipulation" by Igarashi, Moscovich, and Hughes. The purpose of this paper is to create a user friendly interface for manipulating and deforming 2D, mesh-based characters. By completing this assignment, you will also gain experience with posing quadratic optimization problems and optimizing them by solving sparse linear systems (a frequently-occurring theme in computer graphics).

Relevant Reading

- The lecture slides!
- The original paper: As-Rigid-As-Possible Shape Manipulation. Also, check out the website for the paper, which includes a demo video.

Requirements

This assignment is out of **100 points.**

To get full credit (i.e. a grade of A), you must implement the basic algorithm described in the paper. That breaks down as:

- Section 4.1: Scale-free reconstruction **(34 points)**
- Section 4.2.1: Fitting the original triangle to the intermediate triangle **(33 points)**
- Section 4.2.2: Generating the final result using the fitted triangles **(33 points)**

Extra Features

Each of the following features that you implement will earn you extra points. See paper and demo video for more details.

- Depth-Adjustment **(10 points)**
  - Detect overlapping regions dynamically and adjust based on user interaction.
- Vertex Weighting **(10 points)**
  - Add the ability to give vertices added weight to prevent collapse of user-selected regions.
- Custom mesh **(2 points)**
  - Share an interesting custom mesh on Piazza!

- Full image to mesh pipeline **(15 points)**
    - Create a script or equivalent that produces a compatible textured mesh.

Resources

You can download the starter code here: https://github.com/brown-cs-224/Rigid-Stencil
The starter code handles mesh loading, mouse interaction, and rendering. To complete the implementation, you'll need to implement `initDeform` and `deformMesh` in `DeformMesh.cpp`.

You may want to checkout this Eigen page, which contains helpful documentation on Eigen's sparse matrix / sparse linear solver capabilities.

Implementation & Debugging Tips

- We strongly recommend testing out your implementation of Step 1 (scale-free deformation) before moving on to Steps 2 and 3. Since the output of Step 1 is the same as Step 3 (i.e. a deformed mesh), it's quite easy to do this within the code.
- Be sure you really understand the math of each part (in the lecture slides / paper) before you try to translate it into code.
- Most of the challenge of this assignment lies in translating math into code. In particular, a lot of the code you'll write will involve indexing into / reshaping matrices. The Eigen tutorial + the Eigen docs on sparse matrices should cover all of the operations you'll need to write a nice implementation.
- For your reference, we've uploaded the video from class showing how a correct implementation should behave:
    - https://drive.google.com/open?id=1FnVc3I58jA4ZUuEIMxJkgoK8riRd-T16

Submission Instructions

Submit your assignment by running `cs224_handin rigid` from a CS department terminal. You should run the handin script from a directory containing all the files you wish to submit. This directory must include a file named 'README' for the submission to be accepted.