Rigid
CS24 Project 5
Out: Tuesday, March 3
Due: 11:59pm, Thursday, March 19

1 Introduction
In this assignment you are required to implement “As-Rigid-As-Possible Shape Manipulation” by Igarashi, Mosovich, and Hughes. The purpose of this paper is to create a user friendly interface for manipulating and deforming 2D mesh-based characters.

2 Requirements
For a grade of B you are required to implement the base algorithm as outlined in the paper. In particular, your implementation should consist of two main steps:

- Scale free reconstruction (section 4.1)
- Scale adjustment (section 4.2)
- Although not mentioned in the paper you are also required to implement a helper method that prints out the contents of a matrix in human readable format. This method should take a pointer to a matrix and an int that represents the precision of the output (or decimal places is fine), the precision can default to 2 if not specified.

  Important Note: If you come to TA hours for help with the math and do not have this helper method they will not help you until you write it.

For an A grade you may implement one of the suggested extensions:

- Collision detection and depth adjustment
- Weights for controlling rigidity
- Animations

3 Getting Started
A demo implementation is available at: /course/cs224/bin/rigid_demo. Support code can be grabbed from: /course/cs224/asgn/rigid.

The support code handles the process of loading images and triangulating them to create meshes. It also handles all mouse-based interaction and rendering. You'll be implementing the Solver class, in solver.h and solver.cpp. The solver implements the core algorithm. It takes the original mesh and a set of constrained vertices, and produces the new vertex positions by modifying the mesh in-place. You probably won't need to modify any class other than Solver. solver.h contains helpful tips.
Packaged with the support code is a sparse matrix implementation (`SparseMatrix / matrix.h`) and an \( Ax = b \) linear equation solver (`LinearEquation / lineareq.h`). As-Rigid-As-Possible is a linear algebra-heavy paper, so you will likely find these classes useful!

We recommend that you have the mathematics down cold, before writing any code. Read the paper a couple times as well. This will make debugging less painful.

### 4 Handing In

To hand in your program, run `cs224_handin rigid` in a terminal.