CS 184: Computer Graphics and Imaging, Spring 2019

Project 4: Cloth Simulator

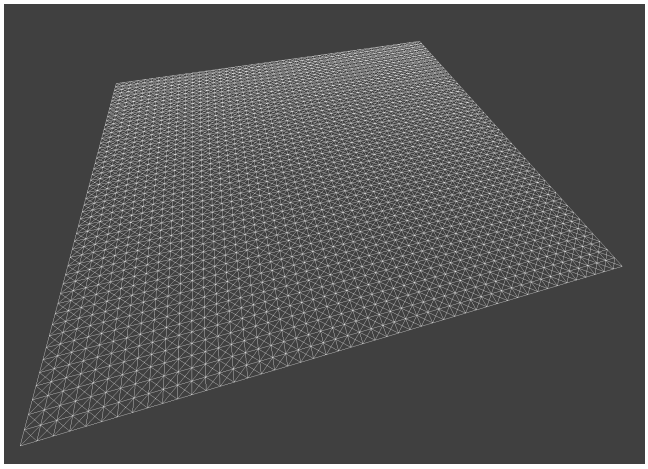
Raymond Ly, CS184-aiv

Overview

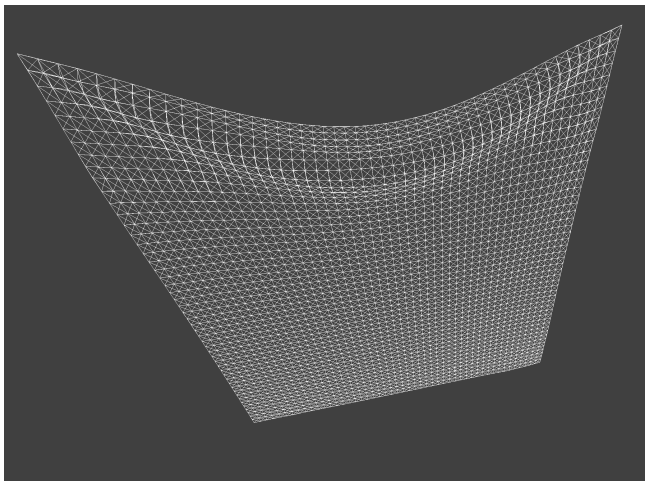
For this project, we sought out to create a realistic cloth simulator that would accurately react to the forces exerted on it. Additionally, we implemented shader functionality to allow our cloth surface to react to lighting, textures, and reflection.

Part I: Masses and Springs

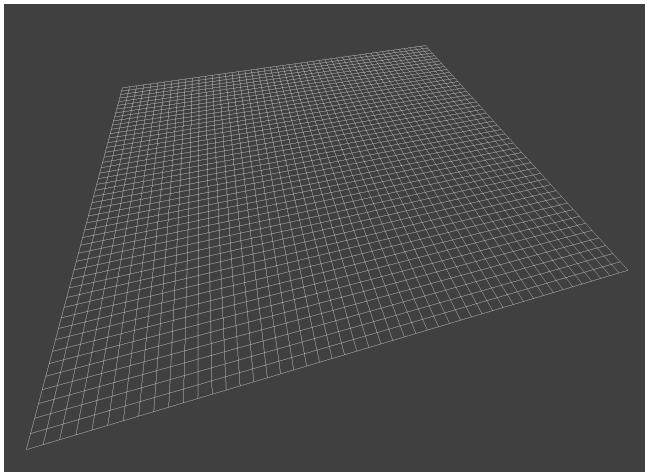
Wireframes:Springs and Point Masses



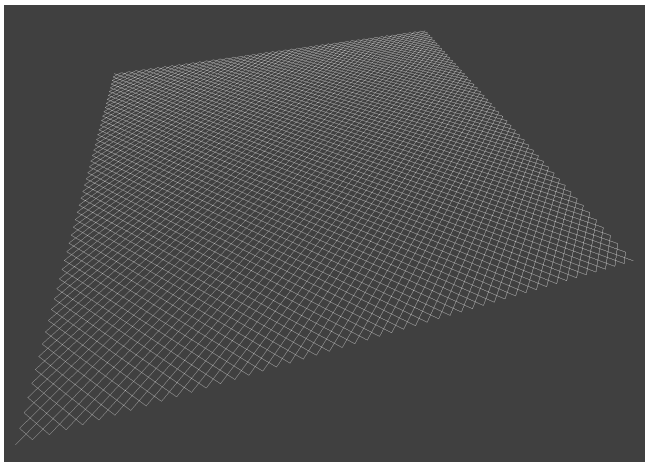
All Springs



All Springs at Rest

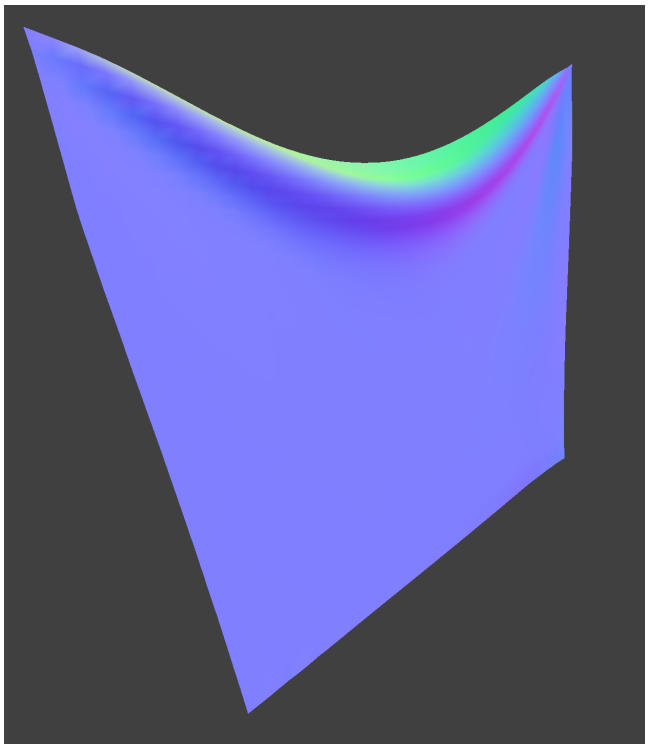


No Shearing Springs



Only Shearing Springs

Part II: Simulation via Numerical Integration

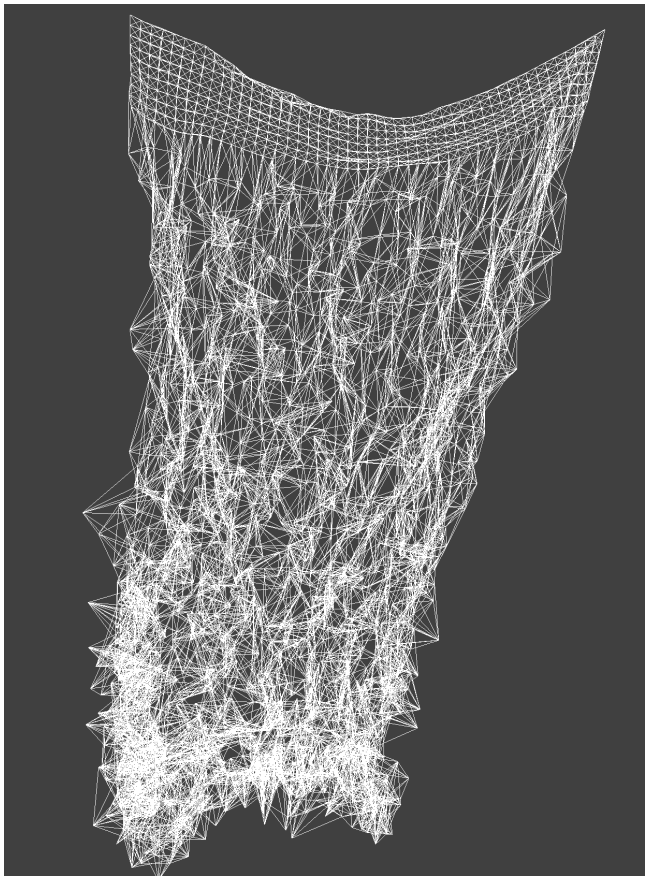


Pinned Cloth at Rest

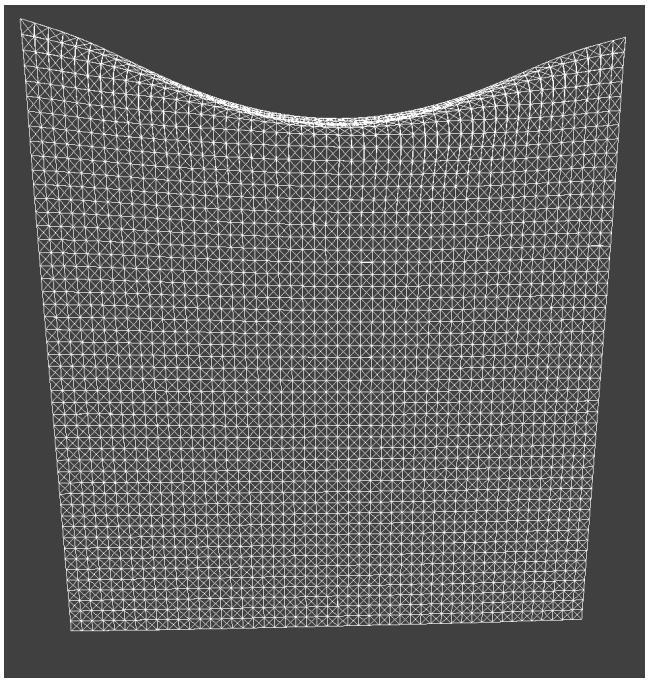
Here, we update our scene to actually be able to simulate the movement of points and interaction of the springs and masses. There are several different variables we may alter to get some interesting results.

Here, we observe that at lower ks the springs tend to deflect drastically. This is die to the fact that ks is a ratio of force to deflection distance. Equally, at a very high ks, there is no deflection at all and the cloth falls in a very rigid fashion.

Spring Interactions at Different ks



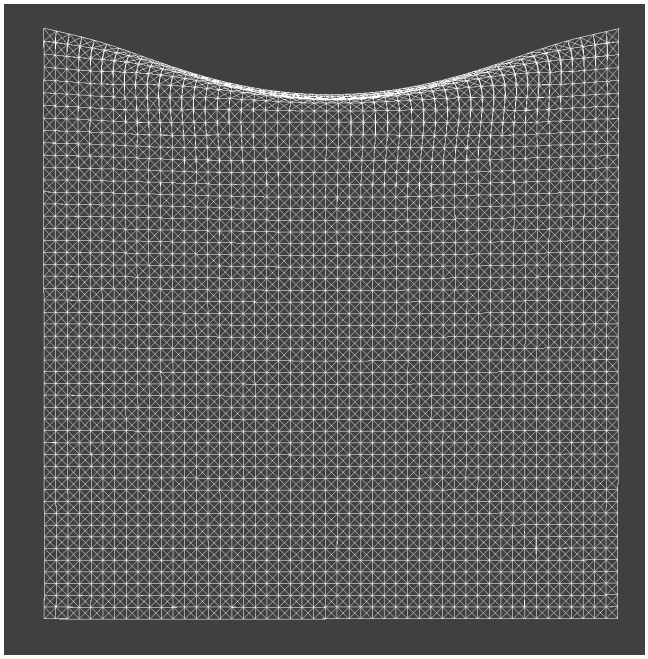
ks = 100 N/m



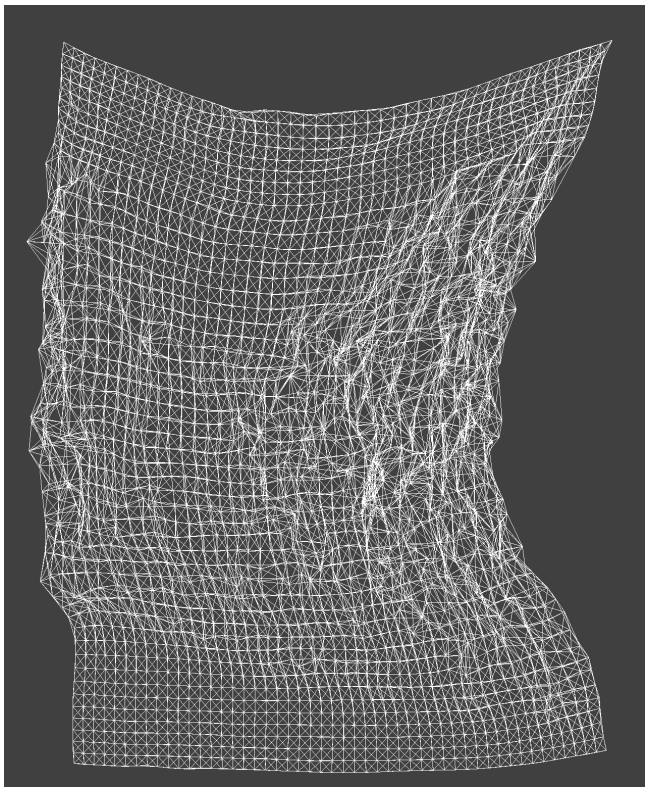
ks = 20000 N/m

By altering the density of our material, we notice the opposite effect. Density is used in our calculation of mass, so the greater the density of some material, the more force it applies in our case , as we use point masses. This force has to be corrected and compensated by the springs in a physical model, so it follows that the greater the density a point mass, the more difficult it becomes for a spring to compensate for its own stretch length.

Spring Interactions at Different ks



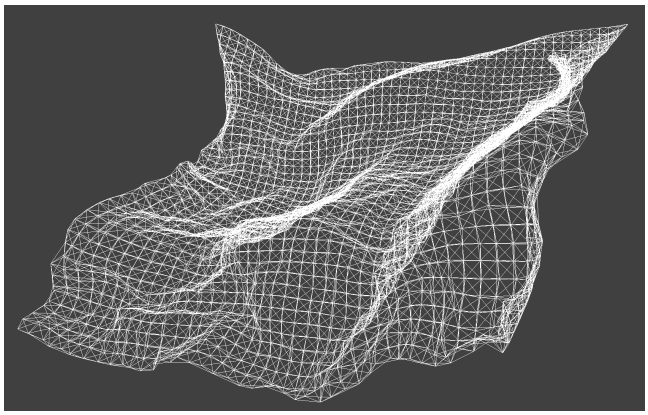
density = 2 g/cm^2



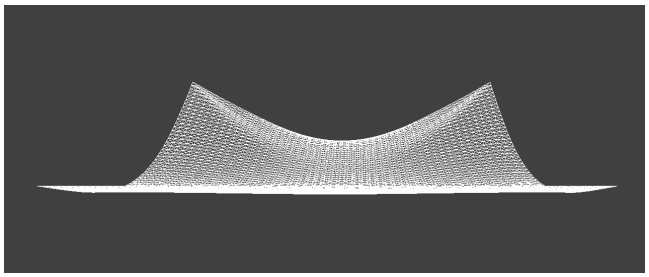
ks = 80 g/cm^2

The last parameter we are interested in between these spring interactions is the damping coefficient. The damping coefficient in spring oscillation is basically a factor that determines the degradation or decay of oscillations in a system. We see this in effect in our images below as the cloth swings back and forth when damping is at 0.0% (i.e. the springs continue to oscillate without degradation) versus a very rigid, restricted movement shown by when our damping factor was maxed out, showing a very slow descent to a resting position.

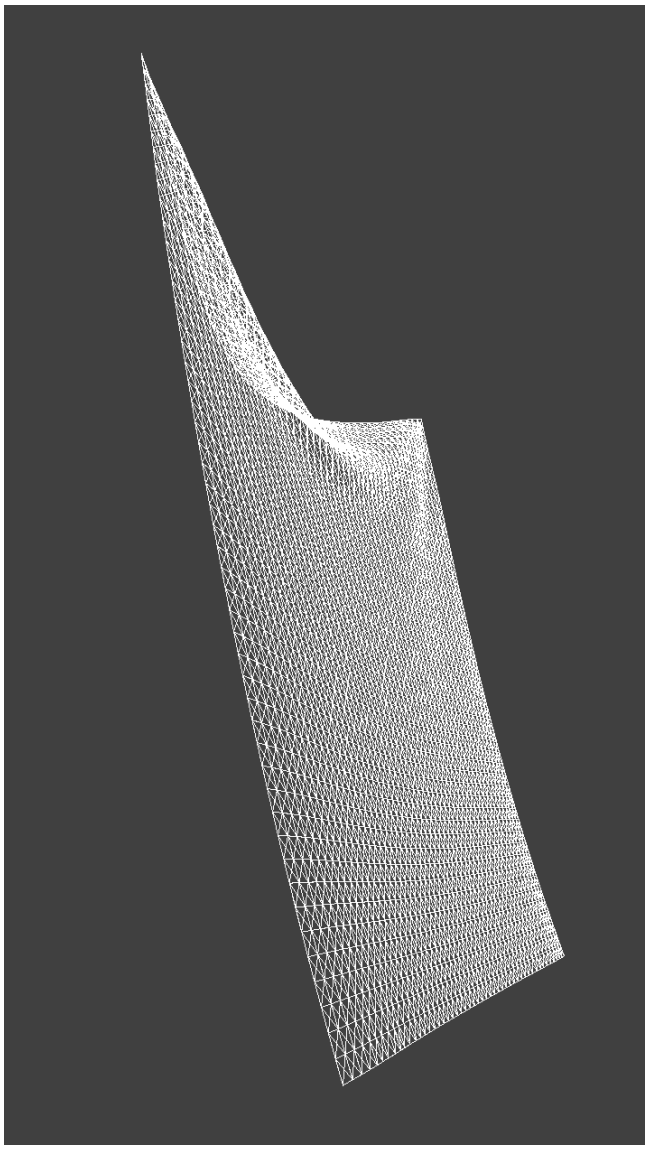
Cloth Collision with a Sphere at Varying Damping



Damping = 0.0%



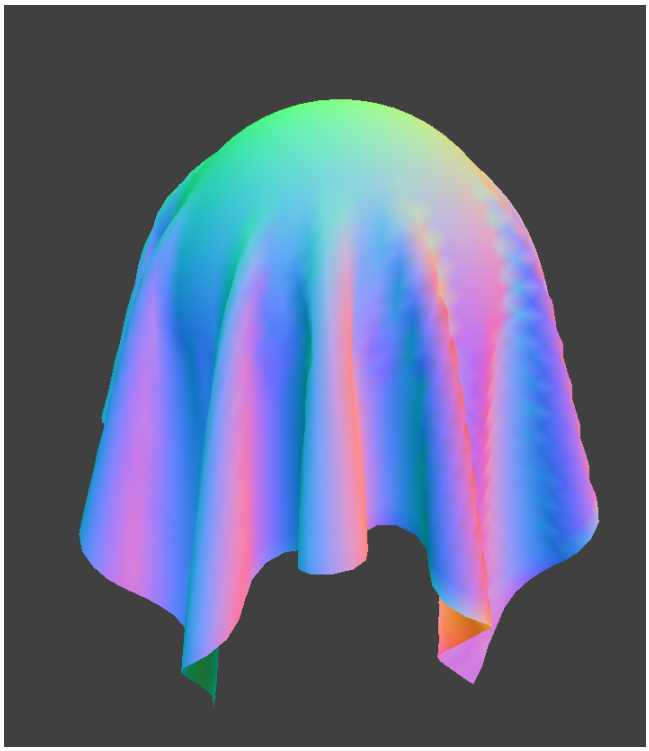
Damping = 1.0%



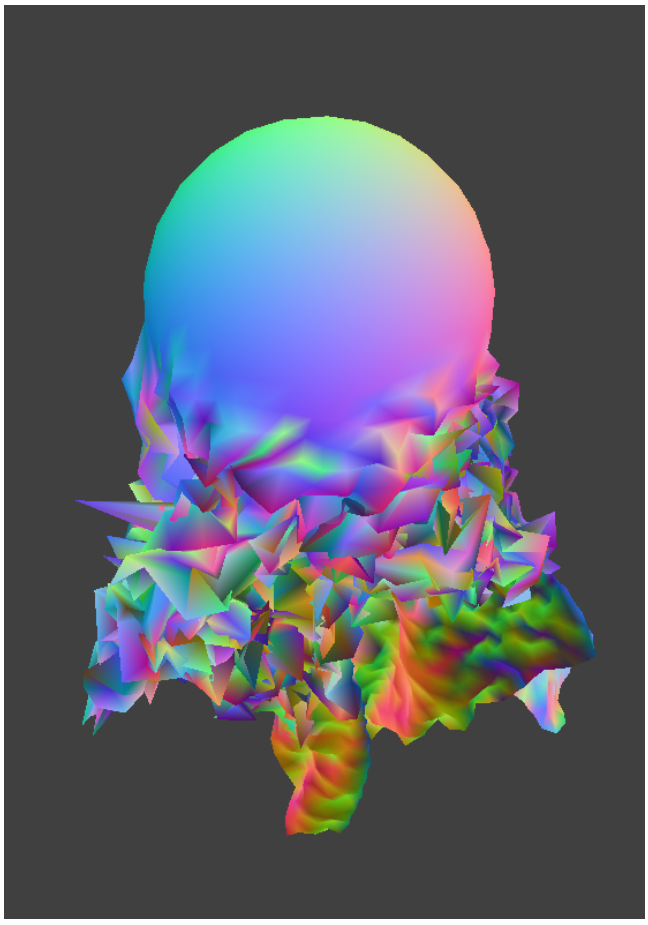
Damping = 1.0% Side View

Part III: Handling Collisions with Other Objects

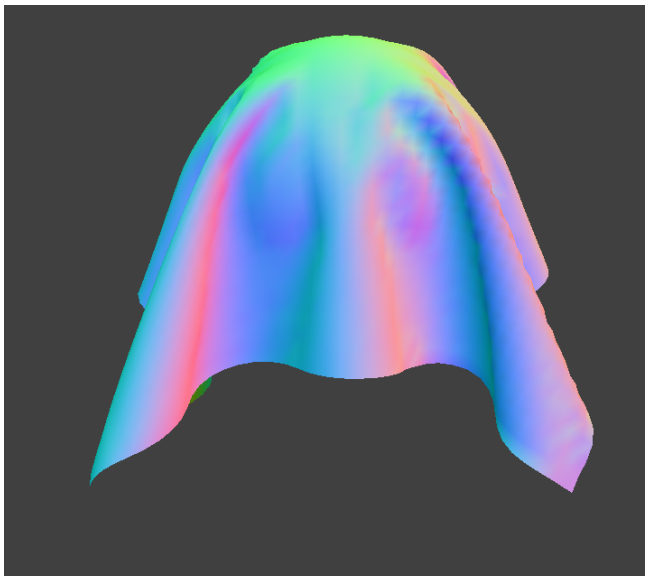
Spring Interactions at Different ks



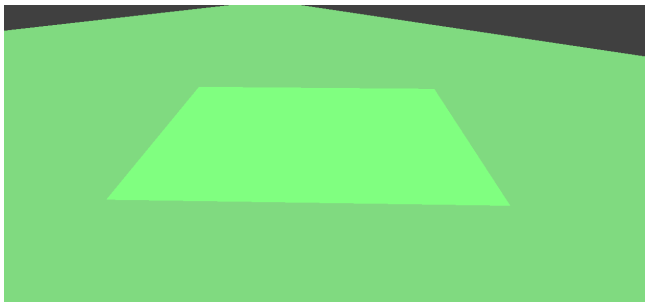
(Default) ks = 5000 N/m



ks = 500 N/m



ks = 50000 N/m

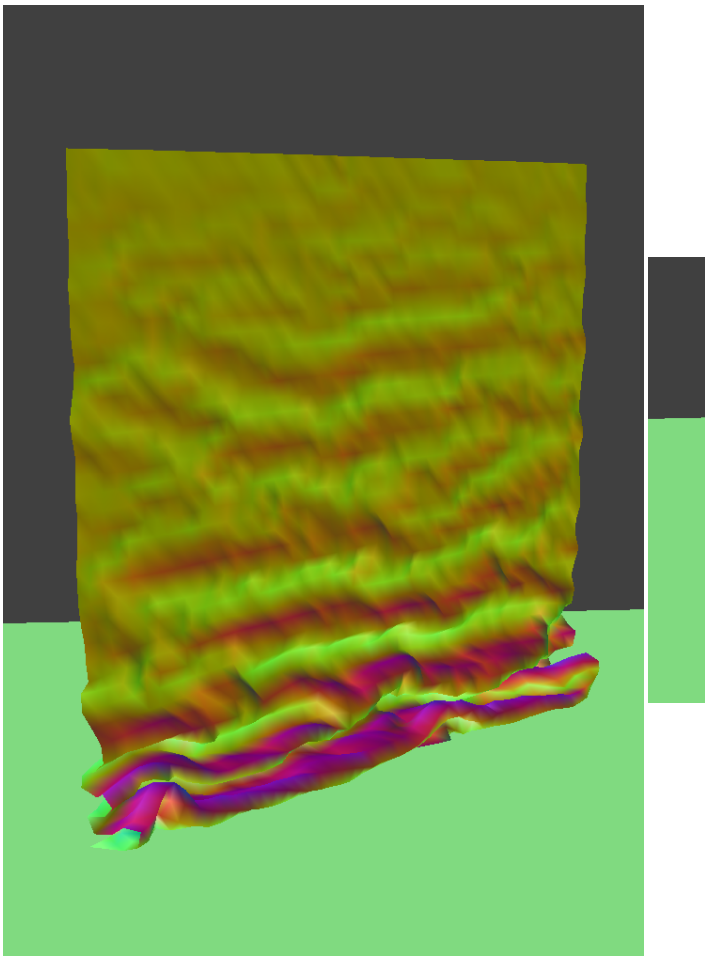


Cloth on a Plane

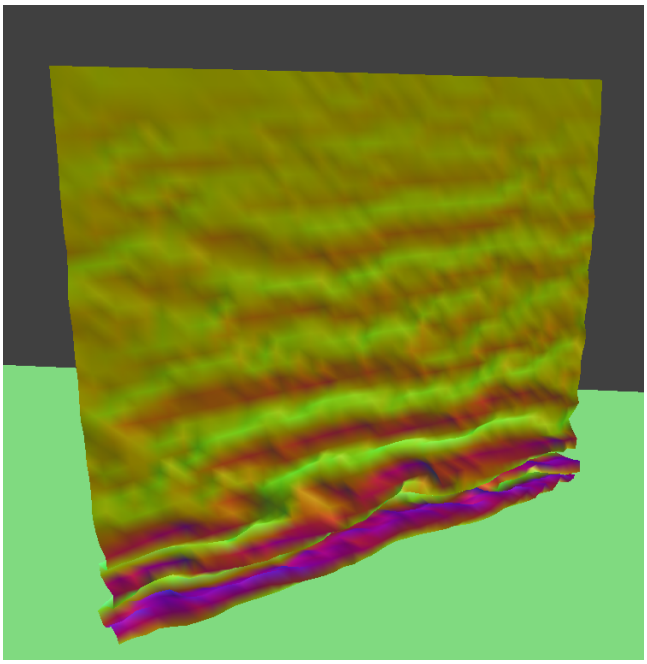
Part IV: Handling Self-Collisions

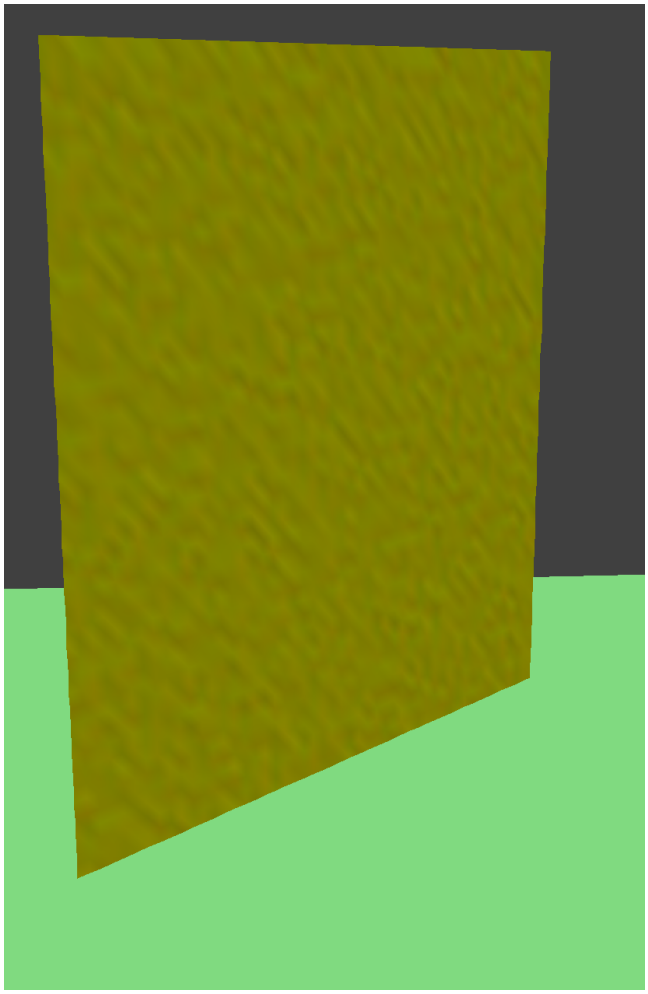
Self Collisions at Varying Parameters

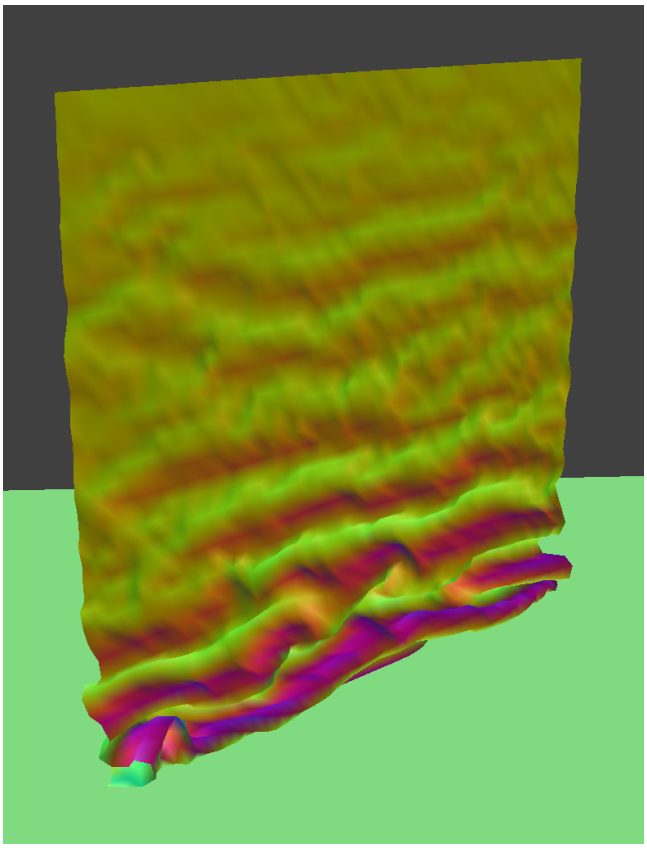
Density = 5 g/cm^2



Density = 20 g/cm^2

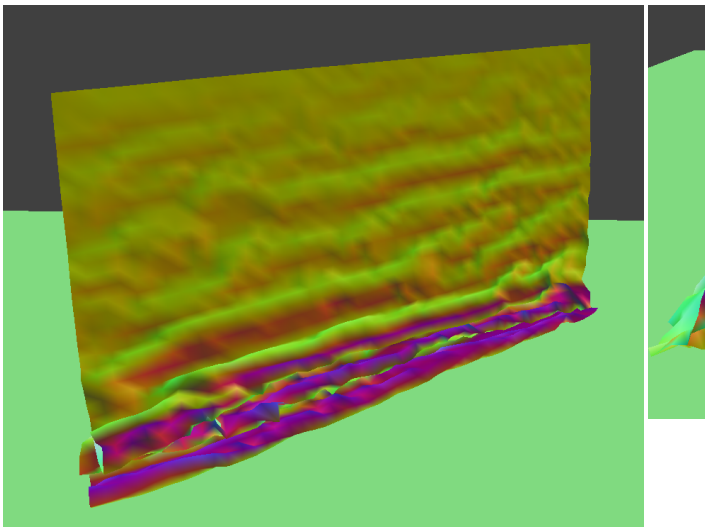


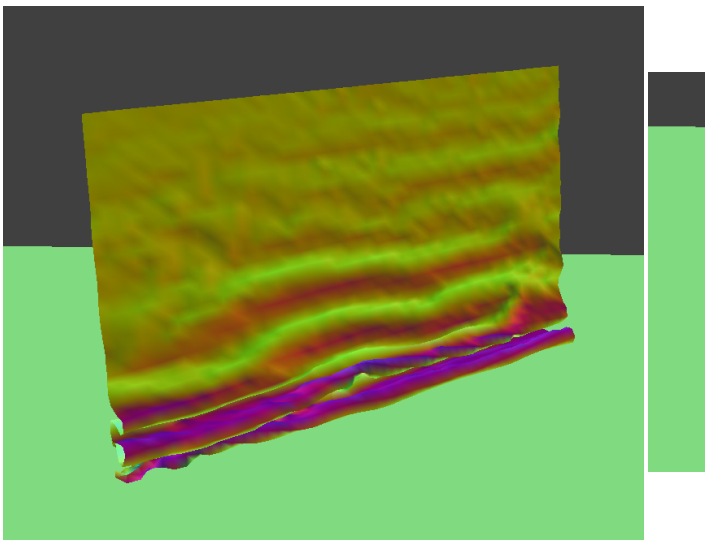




Default Settings: Density = 15 g/cm^2; ks = 5000 N/m

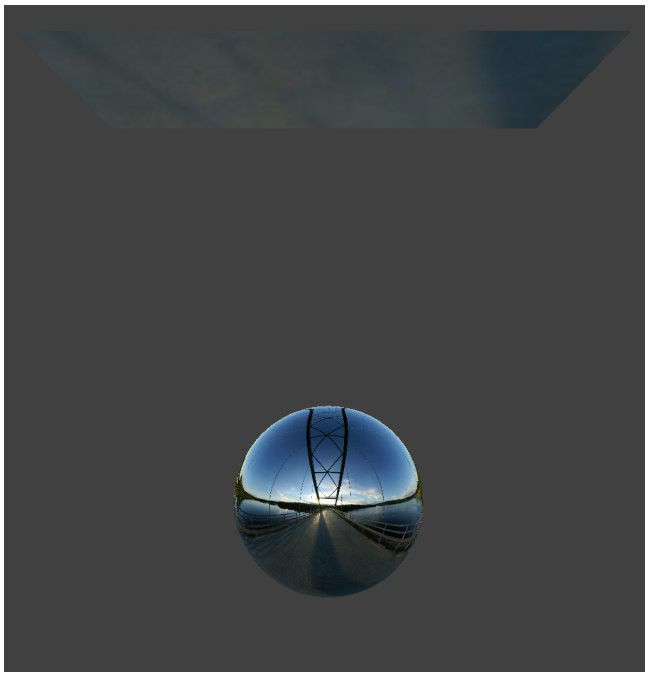
ks = 2500 N/m



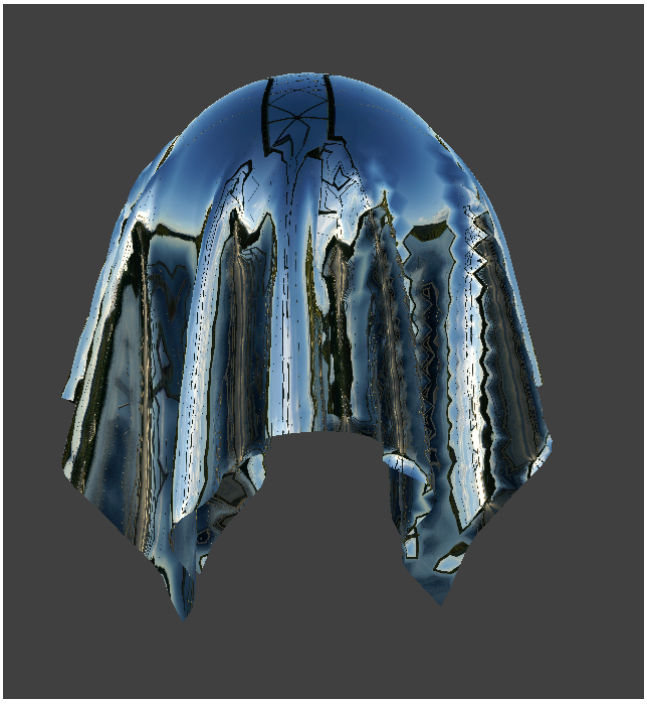


ks = 8000 N/m

Part V: Shaders



Mirror Texture



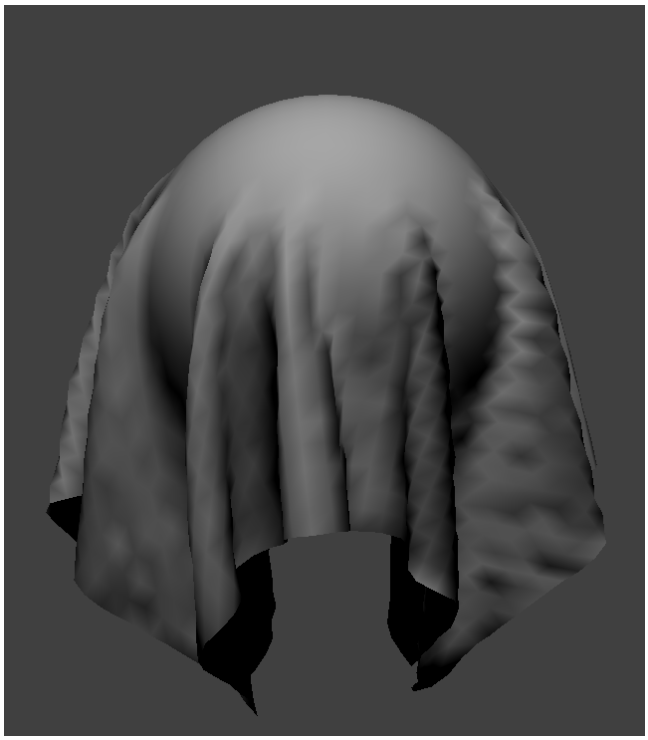
Mirror Texture at Rest

The last part of our project was to implement shaders. Shaders are programs that run in parallel in the GPU that process sections of the graphics pipeline. In effect, a shader is a set of instructions that should be applied to every pixel or portion of some image we are trying to process.

Blinn-Phong shading separates lighting into three levels; ambient, diffuse, and specular. Then each of those channels is summed together for a resultant overal global illumination. This method of lighting allows us to give objects a glossy/shiny surface if we so desire.

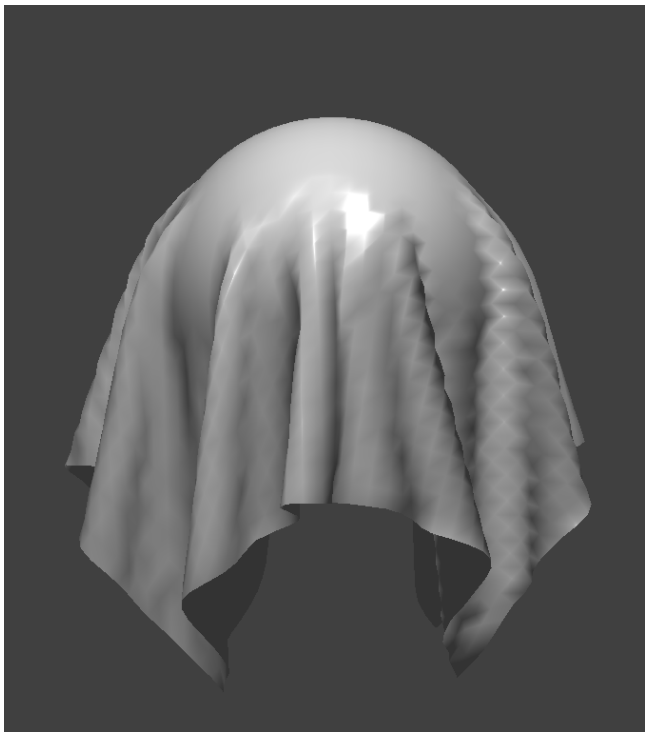
Wireframes:Springs and Point Masses

Ambient Lighting



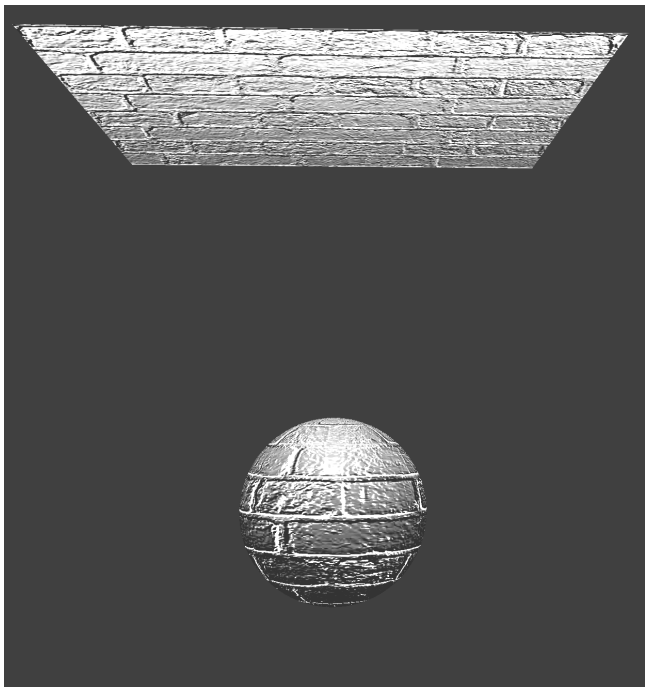
Diffuse Lighting

Specular Lighting

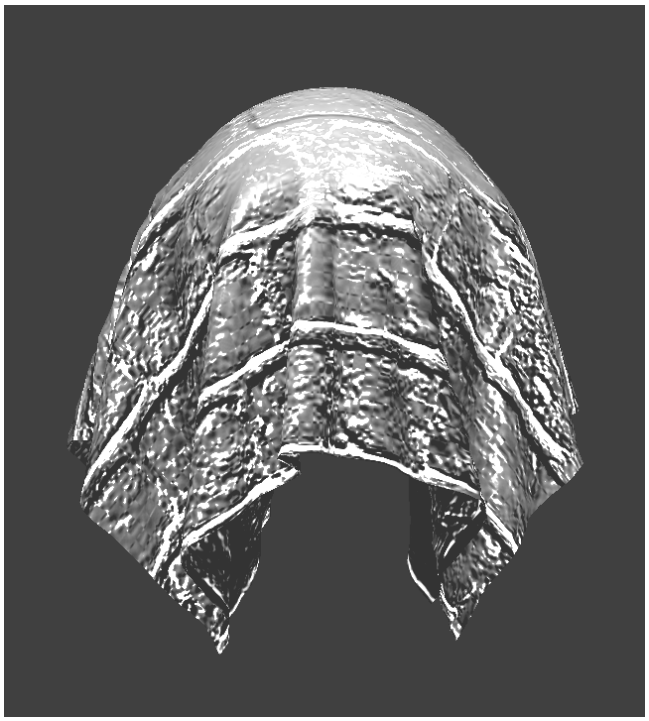


Blinn-Phong Lighting

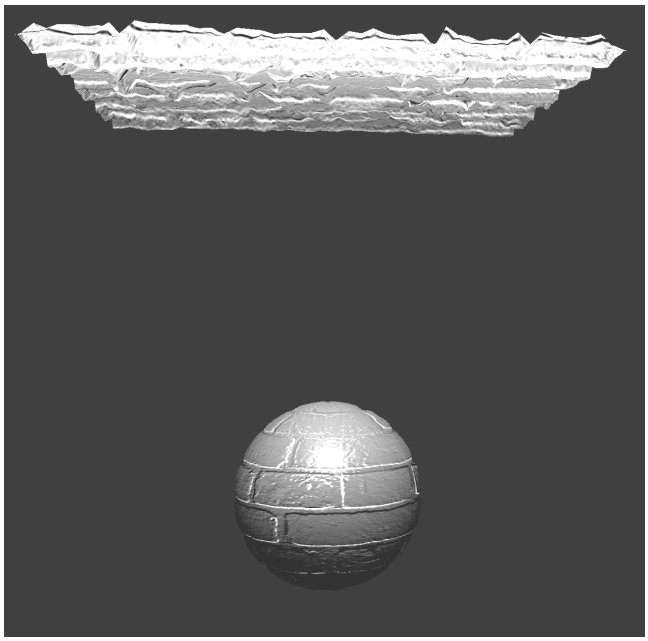
Bump vs Displacement Mapping



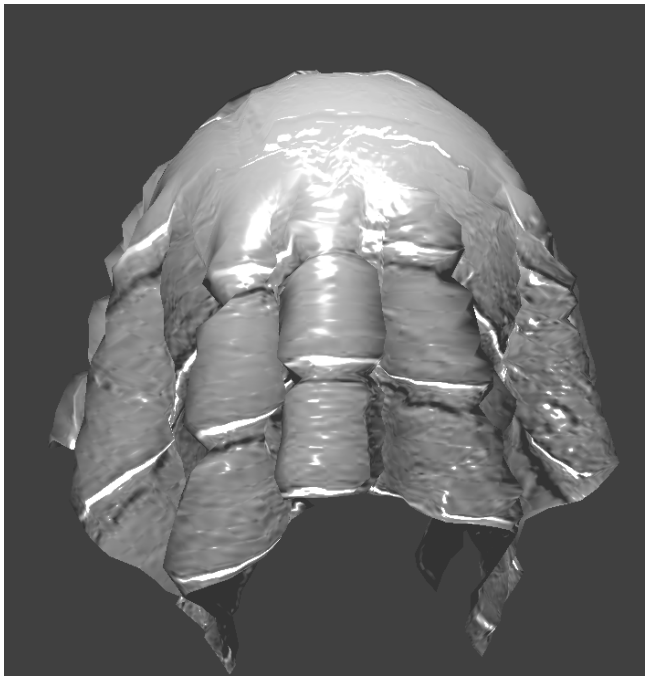
Bump Mapping



Bump Mapping at Rest



Displacement Mapping



Displacement Mapping at Rest