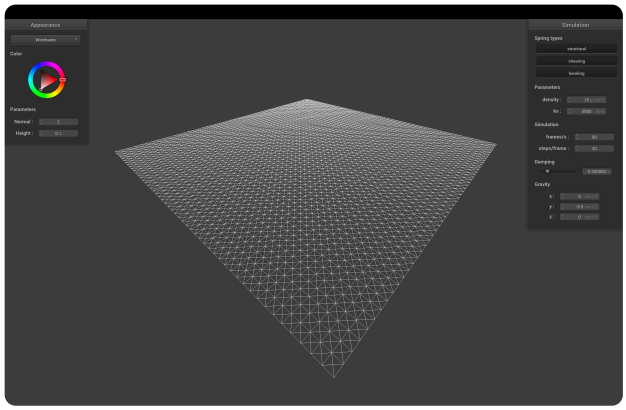
webpage url: https://cal-cs184-student.github.io/hw-webpages-sp24-EmmanuelCobian/hw4/index.html

Homework 4 - Clothsim

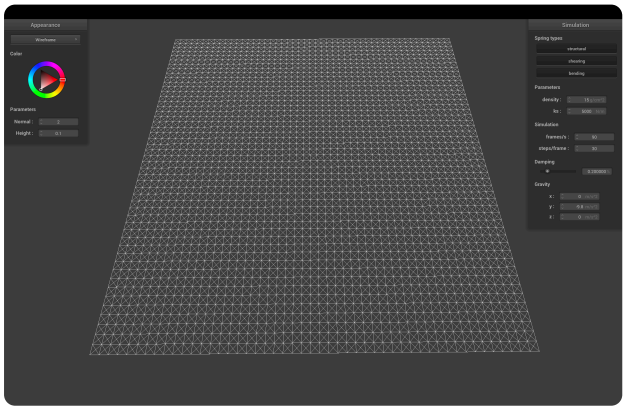
In this assignment I implemented a real-time simulation of cloth using a mass and spring based system. I built the data sctuctures to discretely represent cloth and implemented the physical constratins on them. I also applied numerical integration to simulate the way cloth moves over time. Lastly, I also implemented a collision system that handles both self-collisions and collisions with other objects. One thing that I found really interesting about this project was how the cloth behaved when changing physical constants like spring constants, density, and damping. By changing these basic constants, one gets drastically differnet results in the cloth's behavior.

Part 1: Masses and Springs

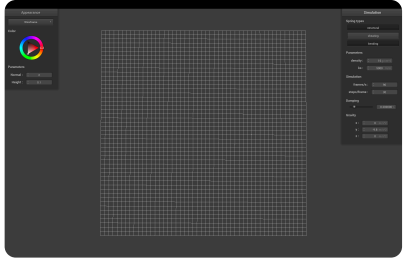
Cloth wireframe examples



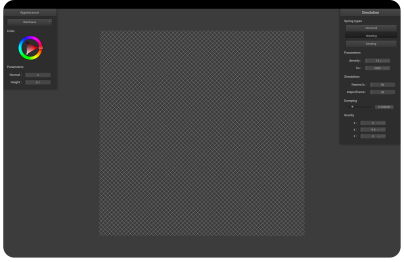
Cloth wireframe example 1



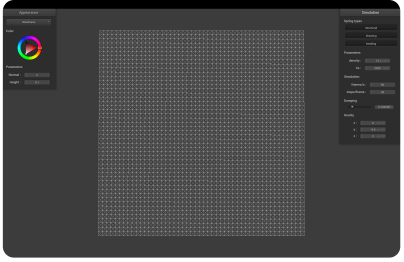
Cloth wireframe example 2



No shearing constraints



Only shearing constraints

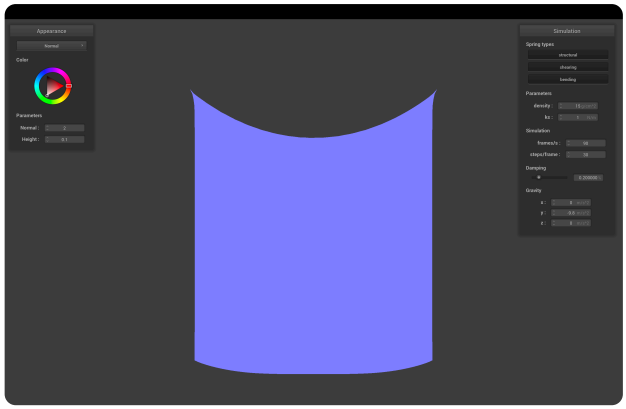


With all constraints

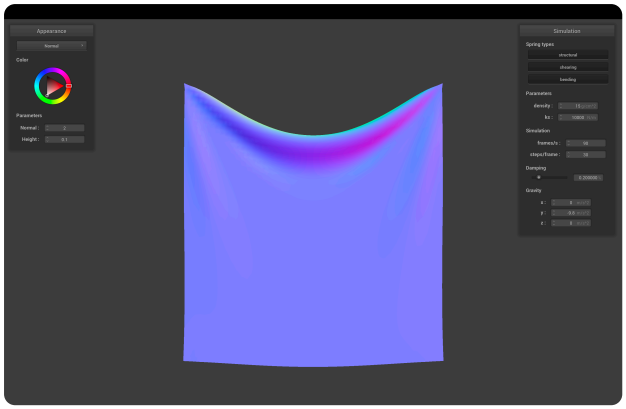
Part 2: Simulation Via Numerical Integration

What are the effects of changing the spring constant `ks`? How does the cloth behave from start to rest with a very low `ks`? A high `ks`?

When using a very low `ks` value, I noticed that the cloth wouldn't wrinkle or fold in any way. It would just fall into steady state like a flat sheet of printer paper. On the other hand, when I used a very high `ks` value, I noticed that the cloth had well defined edges and wrinkles both as it fell into steady state and after it was in steady state.



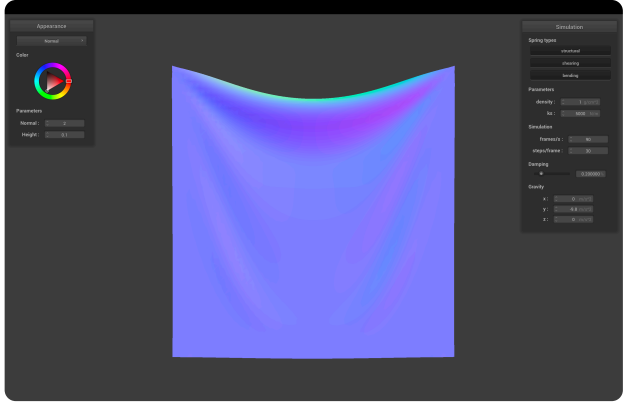
ks = 1



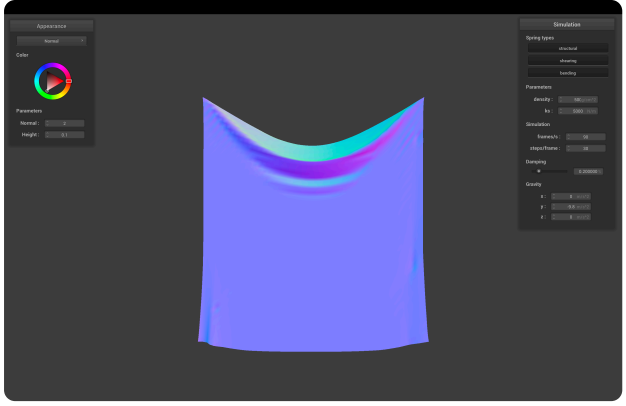
ks = 10000

What are the effects of changing the spring constant `density`? How does the cloth behave from start to rest with a very low `density`? A high `density`?

When using a very low denisty, I noticed that the cloth didn't look all that differnet to the normal parameters; however, I did notice that the top edge of the cloth was more defined and the cloth was more wrinkled. On the other hand, when I used a very high density, I noticed that the top edge of the cloth had multiple wrinkles when compared to the low density settings. The bottom edge is also curved while the low density settings had a straight bottom edge.



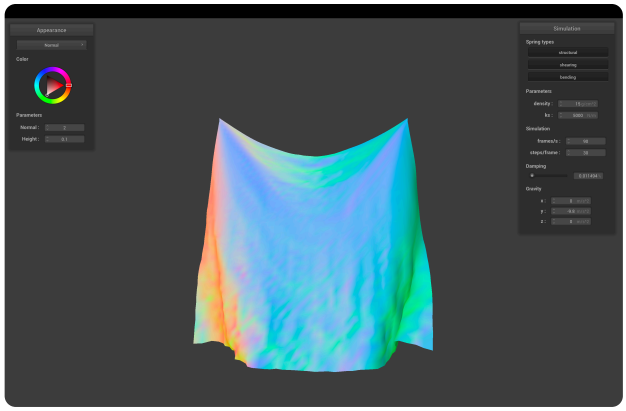
density = 1



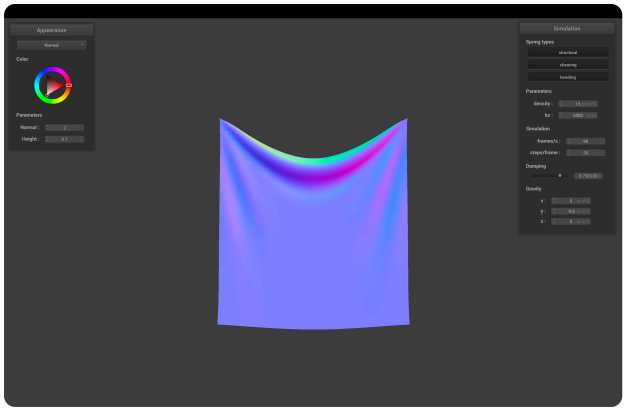
density = 500

What are the effects of changing the spring constant `damping`? How does the cloth behave from start to rest with a very low `damping`? A high `damping`?

When using a very low `damping` value, the cloth flutters about almost uncontrollably, like it's in a wind chamber of sorts. In the image, we see that the cloth folds both towards and away from the camera, something that wasn't seen when playing with other parameters. On the other hand, when using a high `damping` value, I noticed that the cloth falls down slowly and with no fluttering. The cloth is also stiff, no fluttering at all.



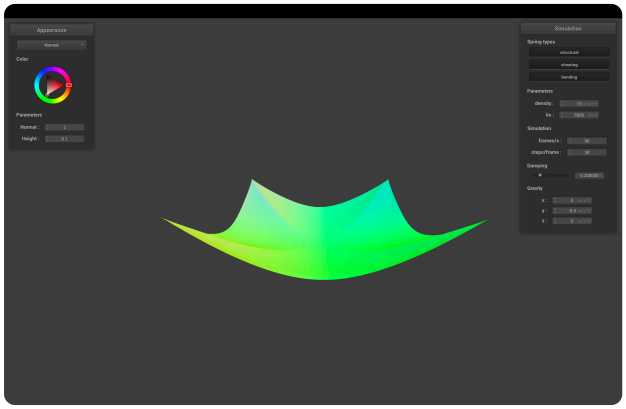
damping = 0.011



damping = 0.79

How does pinned4.json look in its final resting state?

From the image, the cloth with four corners pinned looks like what one would expect. The cloth hangs and the middle of the cloth stoops down a bit.



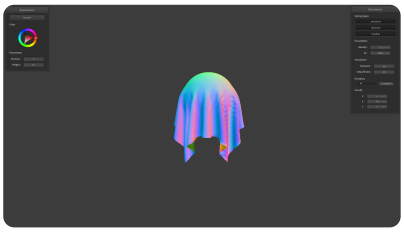
pinned4.json

Part 3: Handling Collisions with Other Objects

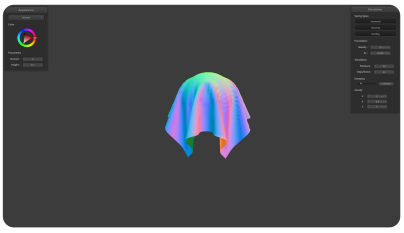
How does the cloth over a sphere look like with different `ks` values?

When using small values of `ks`, I notice that the cloth droops down low; the cloth has more surface area contact with the sphere when compared to the default value of `ks`. On the other hand, when using a large value of `ks`, I notice that the cloth appears more bulky in a sense. When the cloth falls onto the sphere, the cloth folds and makes it seem like it's more stiff. Less of the cloth's surface touches the sphere when compared to a low value of `ks`.

ks = 500



ks = 5000



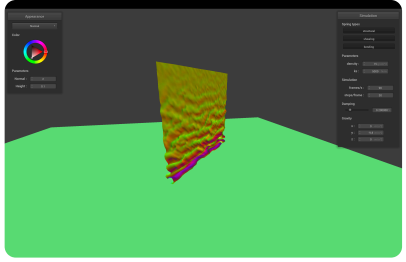
ks = 50000

How does the shaded cloth look like at rest on the plane?

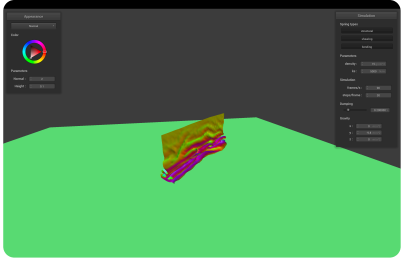
Cloth on plane at rest

Part 4: Handling Self-Collisions

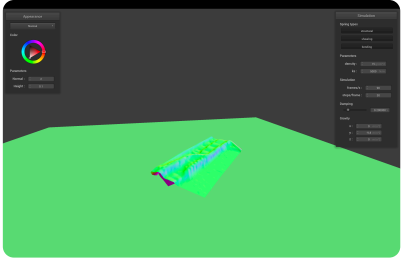
What does the cloth look like at different stages of falling?



Early state



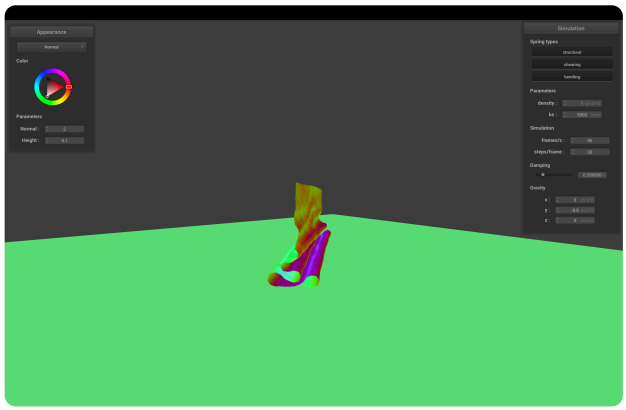
Mid state



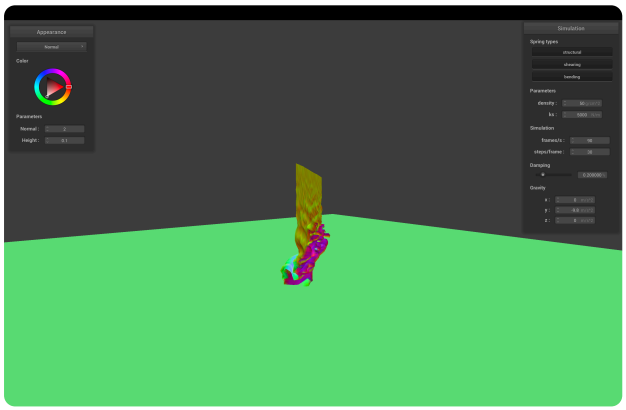
End state

How does the simulation change when altering `density` and `ks` values?

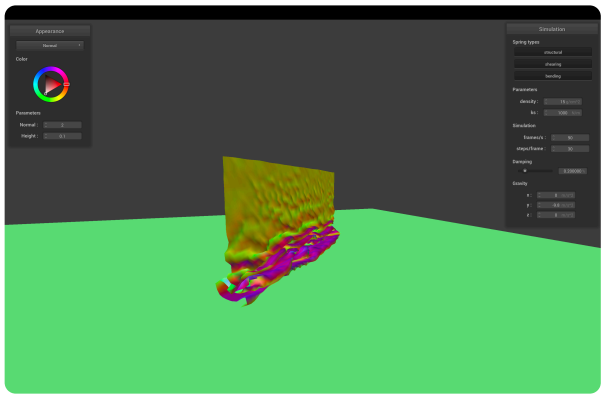
When density is low, the cloth falls like a sliky material. The cloth also doesn't fold on itself as much when compared to when density is high. Furthermore, when the density is high, the cloth has many more folds on itself and takes longer to reach steady state. When the ks constant is low, I notice that, similar to when density is high, the cloth has many more folds and takes a bit longer to reach steady state because of the folds in the cloth. On the other hand, when ks is high, the cloth doesn't have as many folds and it falls more smoothly; the cloth doesn't take long to reach steady state.



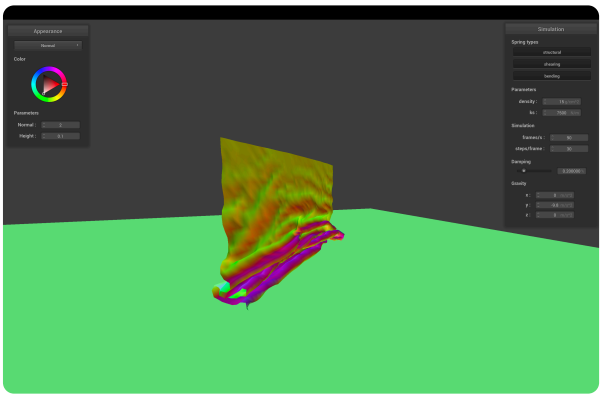
density = 1



density = 50



ks = 1000



ks = 7500

Part 5: Shaders

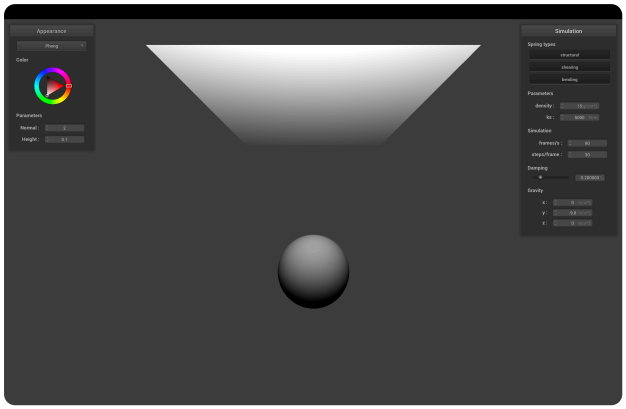
What is a shader program and how do vertex and fragment shaders work together to create lighting and material effects?

Shader programs are isolated programs that run in parallel on the GPU, meaning that we can accelerate computations and display lighting effects for real-time and interactive applications. Shader programs execute sections of the graphics pipeline by taking in some input and outputting a single 4D vector. Vector shaders transform and change gemoetric properties of vertices. These transformations become the inpuut for fragment shaders that then compute a color using the transformations.

What is the Blinn-Phong shading model?

The Blinn-Phong shading model is a method used to calculate the illumination of surfaces. It combines three different types of lighting effects, ambient, specular, and diffuse, to produce one cohesive output. In the end, this model determines the color of each pixel based on its orientation relative to light sources and the viewer's position, helping create a realistic lighting effect.

Ambient component only



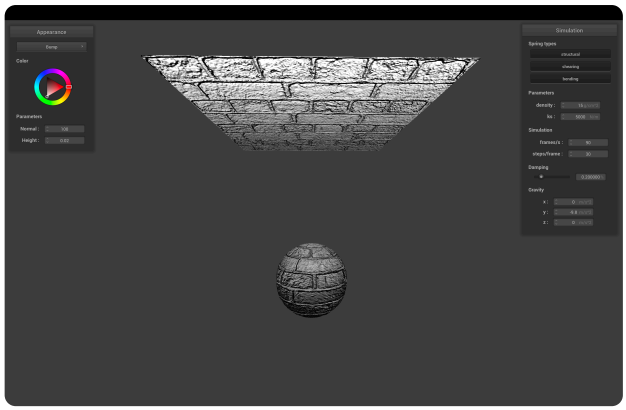
Diffuse component only

Specular component only

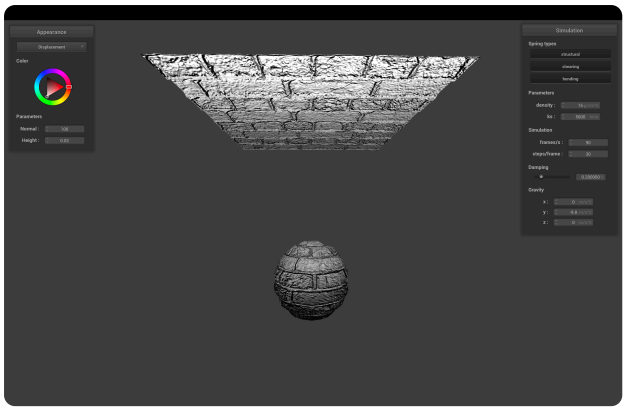
Complete blinn-phong model

Bump vs displacement mapping with different mesh coarseness

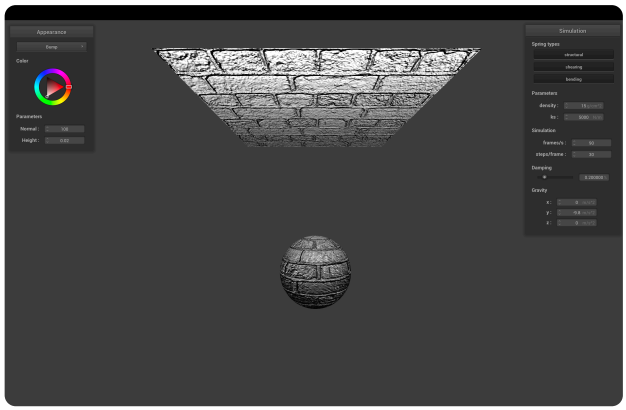
Bump mapping modifies normal vectors of an object such that the fragment shader gives the illusion of detail. On the other hand, displacement mapping is the modification of the vertices' positions to reflect the height map on top of also changing the vector normals. On the renderes, I notice that bump mapping does a good job of making the texture look like it's "popping out" in a sense. On the other hand, the displacement render does a good job of accentuating the bumps and making them stick out through the change in vertex positions. Finally, when the coarseness is set with `-o 16` and `-a 16`, the texture doesn't wrap around the sphere as smoothly; I notice jagged edges here and there. On the other hand, when the coarseness is set with `-o 128` and `-a 128`, the texture smoothly wraps around the sphere and differences in vector position come only from the implementation of displacement mapping.



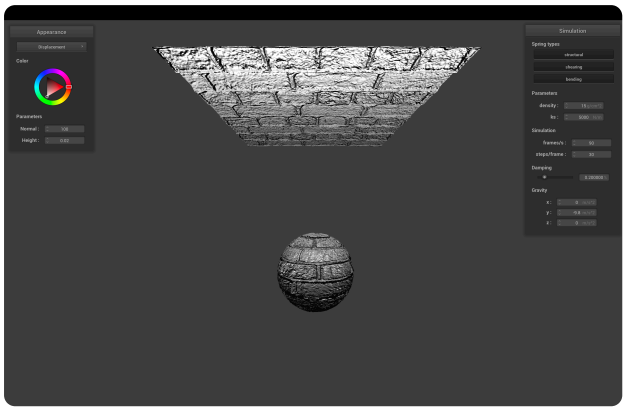
Bump mapping -o 16 -a 16



Displacement mapping -o 16 -a 16

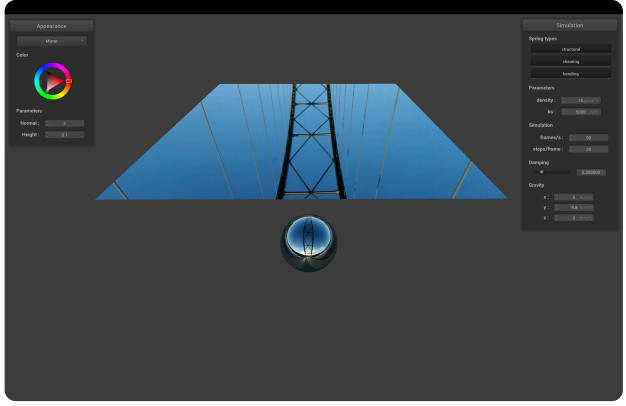


Bump mapping -o 128 -a 128

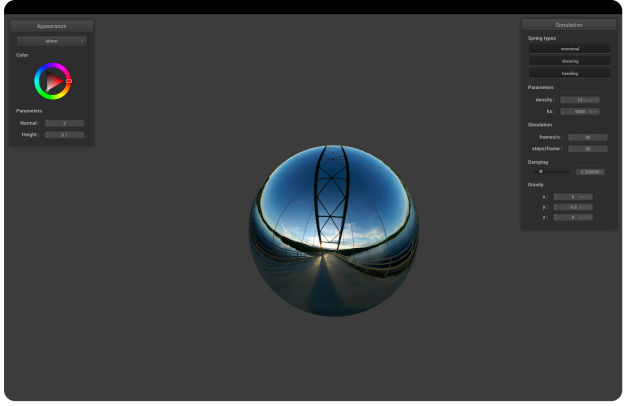


Displacement mapping -o 128 -a 128

Screenshots of a mirror shader on the cloth and on the sphere



Mirror shader on cloth



Mirror shader on the sphere

Emmanuel Cobian Duarte - CS 184 - Spring 2024