CS184 Homework 4: Cloth Sim

Martin Guo

guozy@berkeley.edu

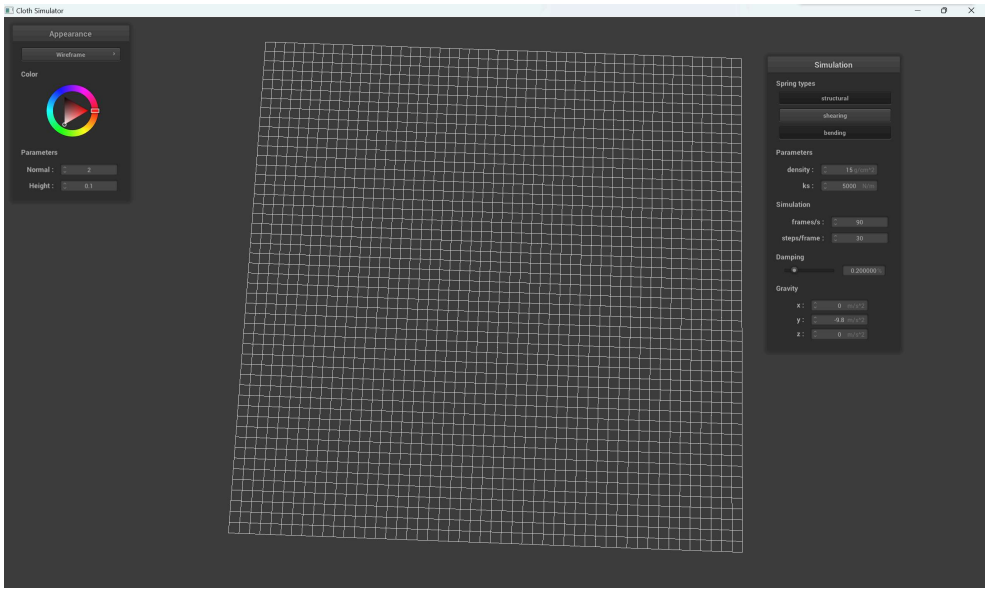
This webpage is available at: https://cal-cs184-student.github.io/hw-webpages-sp24-0-0-00-0/hw4/index.html

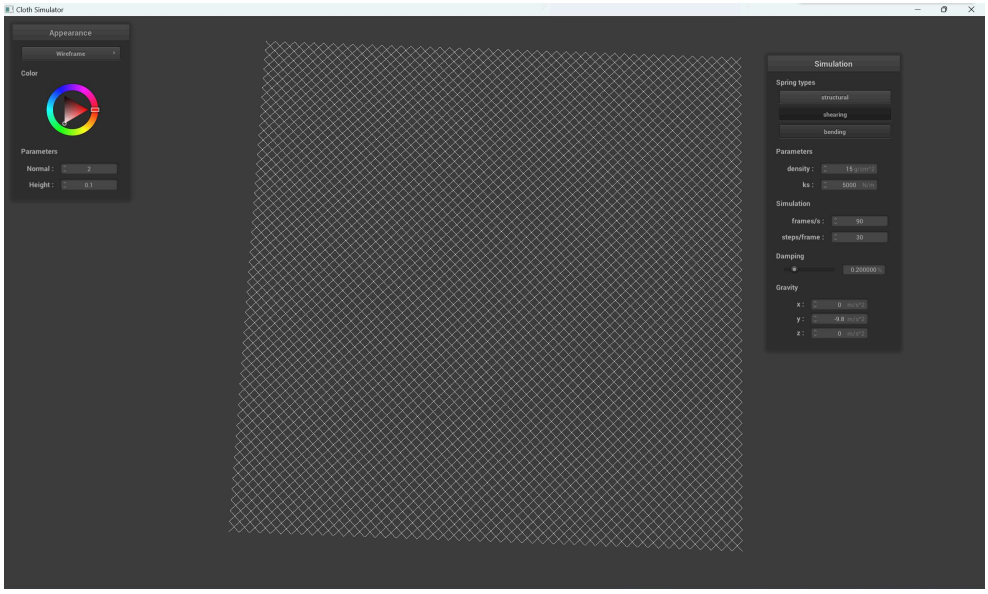
Overview

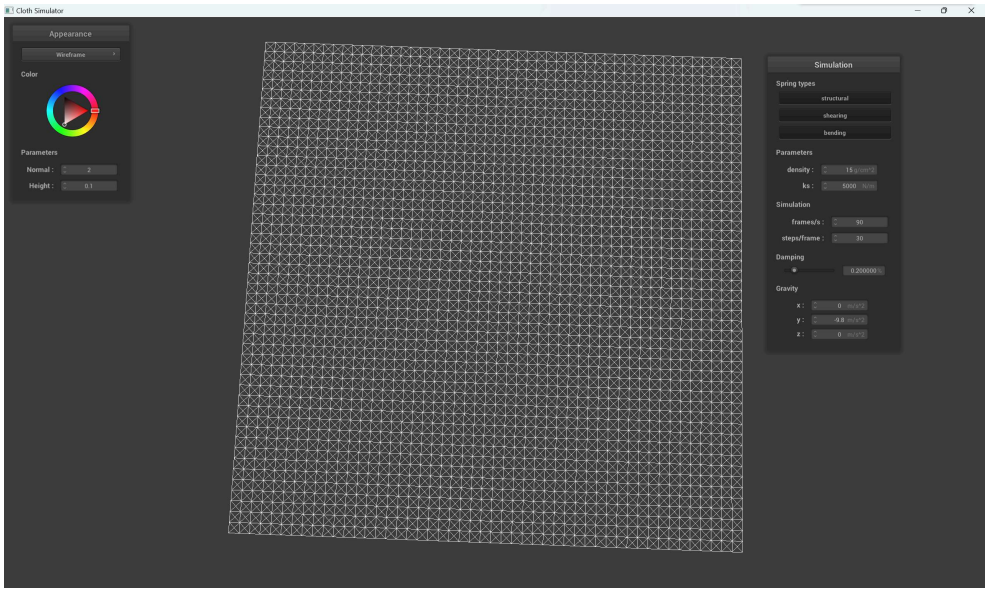
In this homework I implemented the masses-and-springs representation of a cloth, implemented physical simulation using forward Euler method and Verlet integration, realized collisions with objects and collisions with the cloth itself, and wrote a number of simple shaders to display the cloth while simulating.

Part 1: Masses and Springs

The following are masses and springs displays for `scene/pinned2.json` without shearing constraints, with only shearing constrains, and with all constraints respectively.





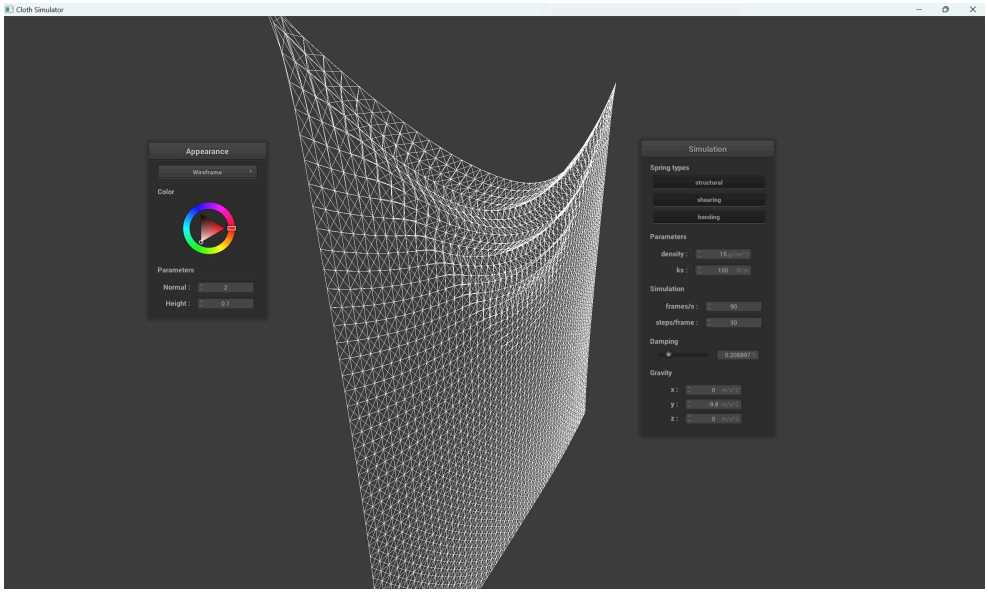


Part 2: Simulation via numerical integration

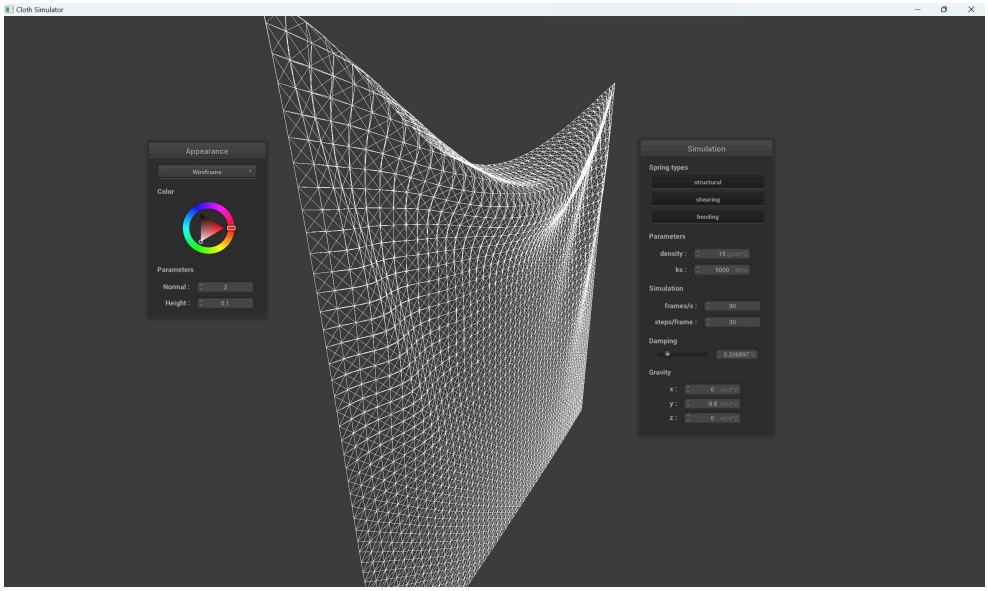
Various parameter changes are experimented with the cloth simulation on `pinned2.json`. Parameters involved include `ks`, `density` and `damping`.

1. With high `ks`, the cloth is more strectchy and tend to curl up together; with low `ks`, the cloth tends to be more expanded and more easily folded. The final resting state of low `ks` has more creases in the top of the cloth than that of high `ks`, and the creases are shallower.

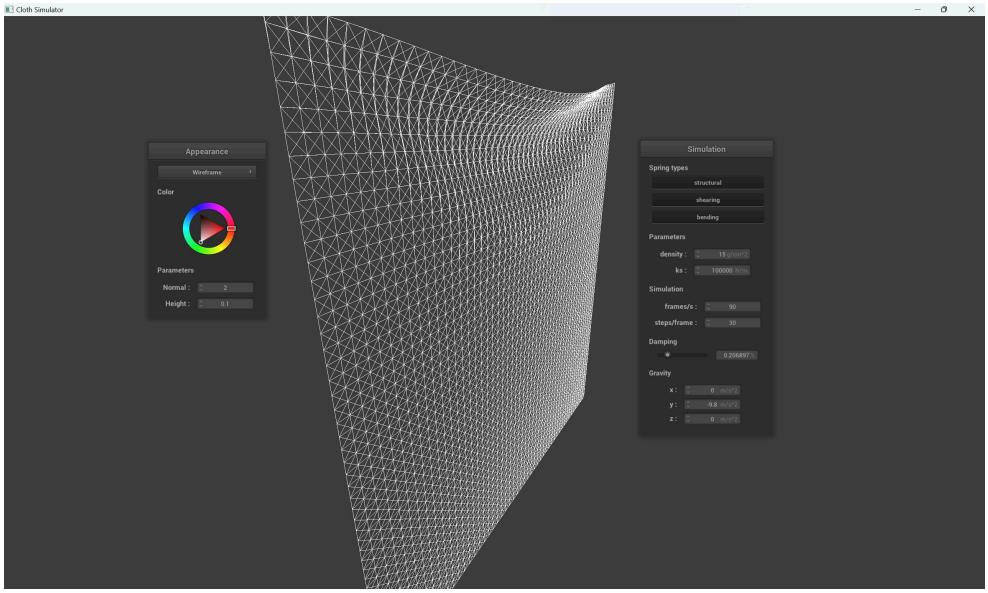
Below are some final resting states of `pinned2.json` with varying `ks`. Other parameters are default.



ks = 100



ks = 5000 (default)

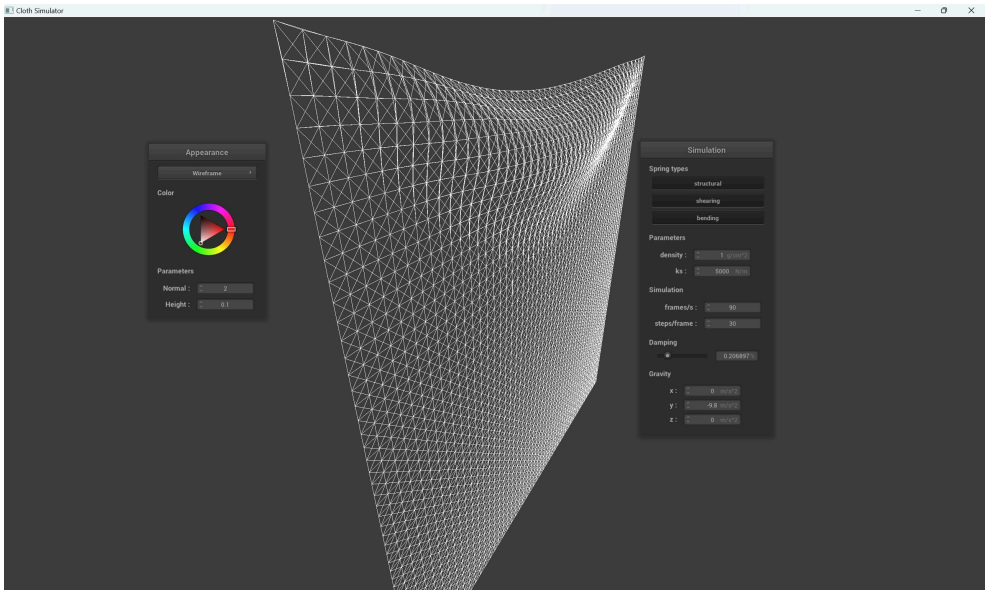


ks = 100000

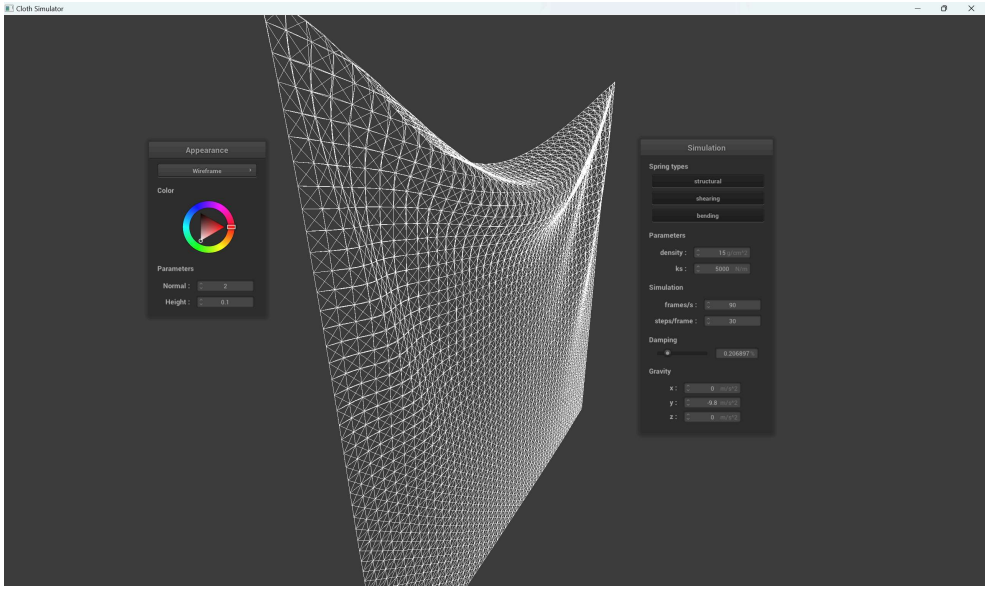
2. With high `density`, the cloth is much heavier and more draped downwards. With low `density`, the cloth has more tendency to drift in the air.

Also, if `ks` and `density` are scaled by the same amount, the final resting state are the same.

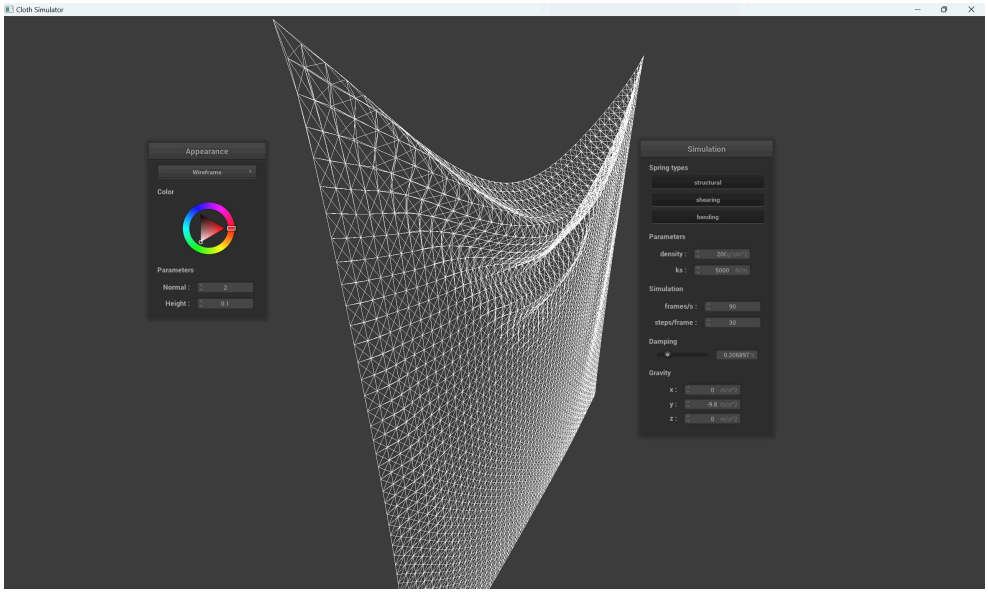
Below are some final resting states with varying `density`. Other parameters are default.



`density` = 1



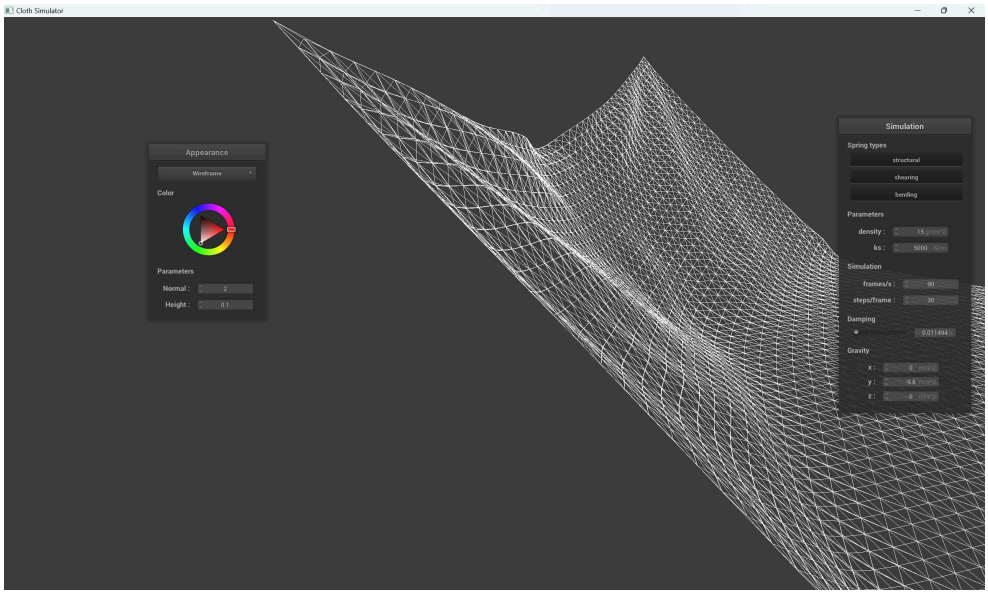
`density` = 15 (default)



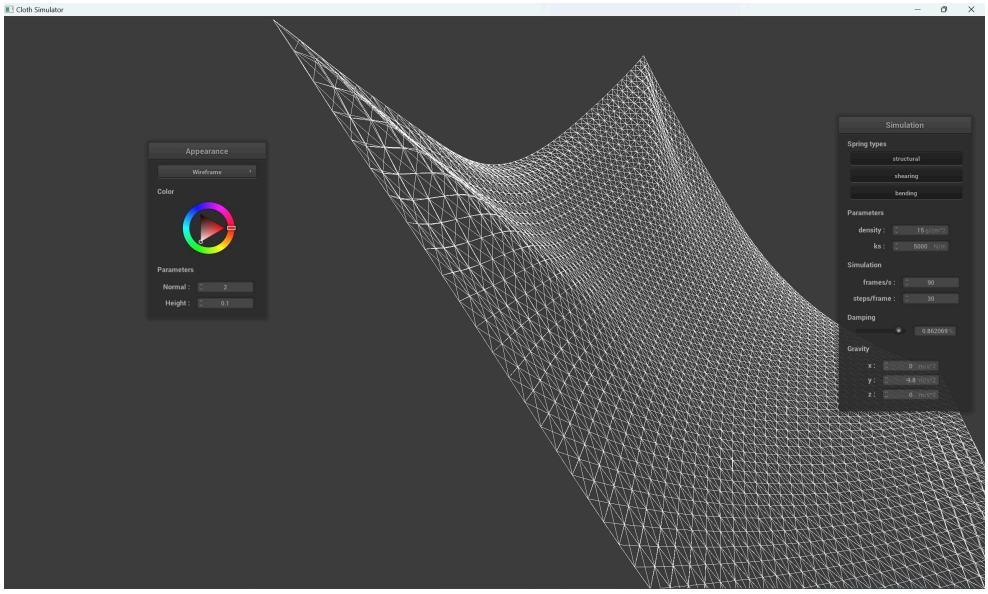
`density` = 200

3. The `damping` parameter acts as the resistance on the cloth. It doesn't make any noticeable difference on the final resting state, but when `damping` is high, the cloth falls much slowly as though in thick liquid; when `damping` is low, the cloth moves much quicker and more freely. When `damping` = 0, the cloth doesn't come to a rest after a very long time.

Screenshots of the cloth in the process of falling from starting position, with varying `damping` , other parameters are default:

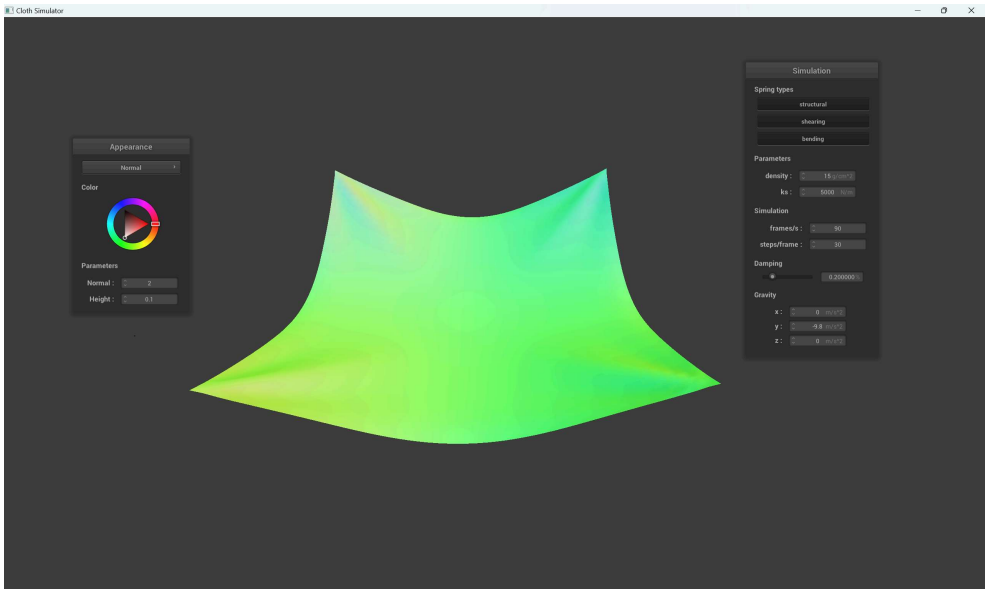


`density` = 0.011494



`density` = 0.862069

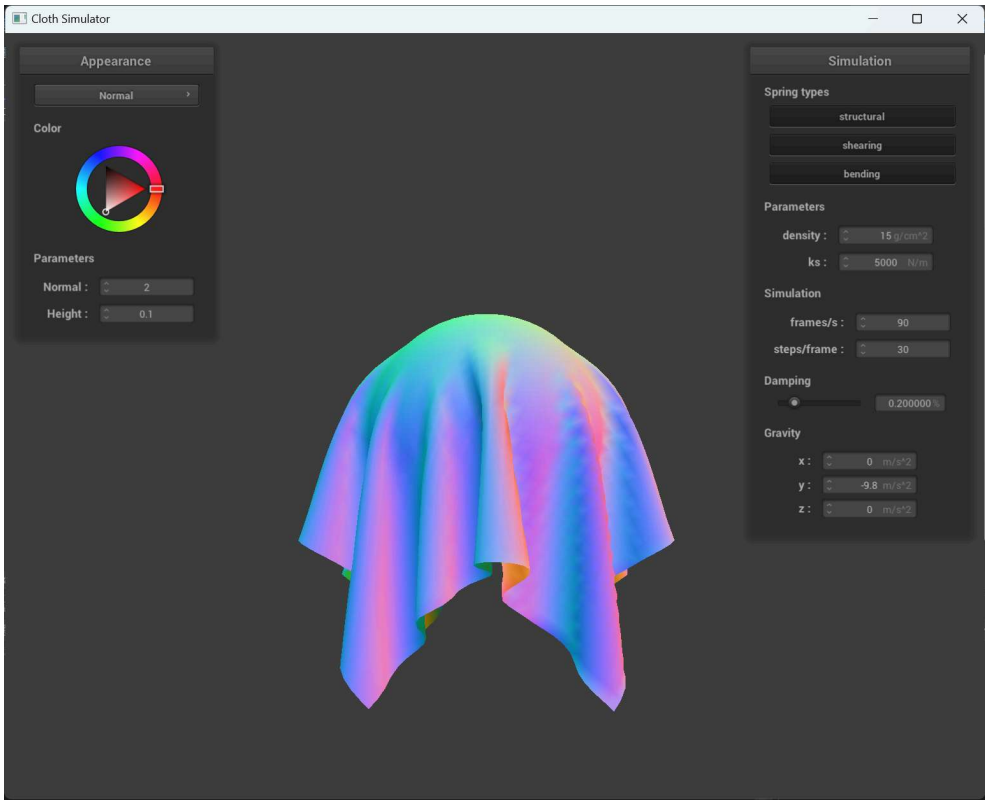
Here's an extra screenshot from `pinned4.json` in its final resting state, with default parameters and normal appearance.



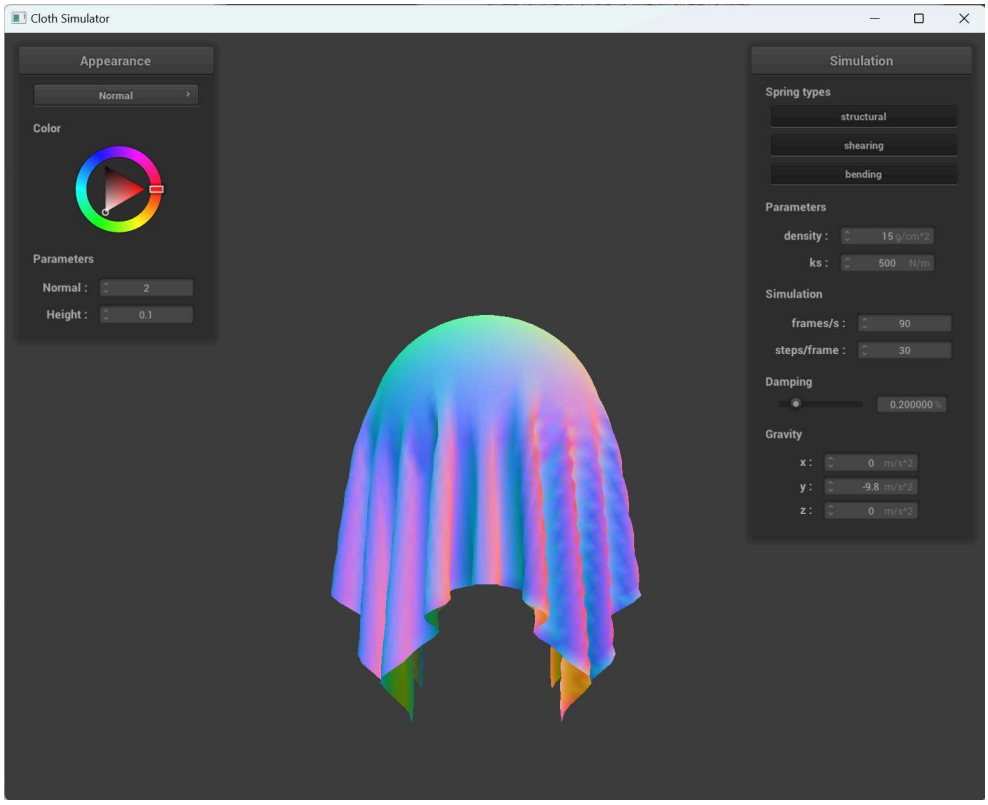
Part 3: Handling collisions with other objects

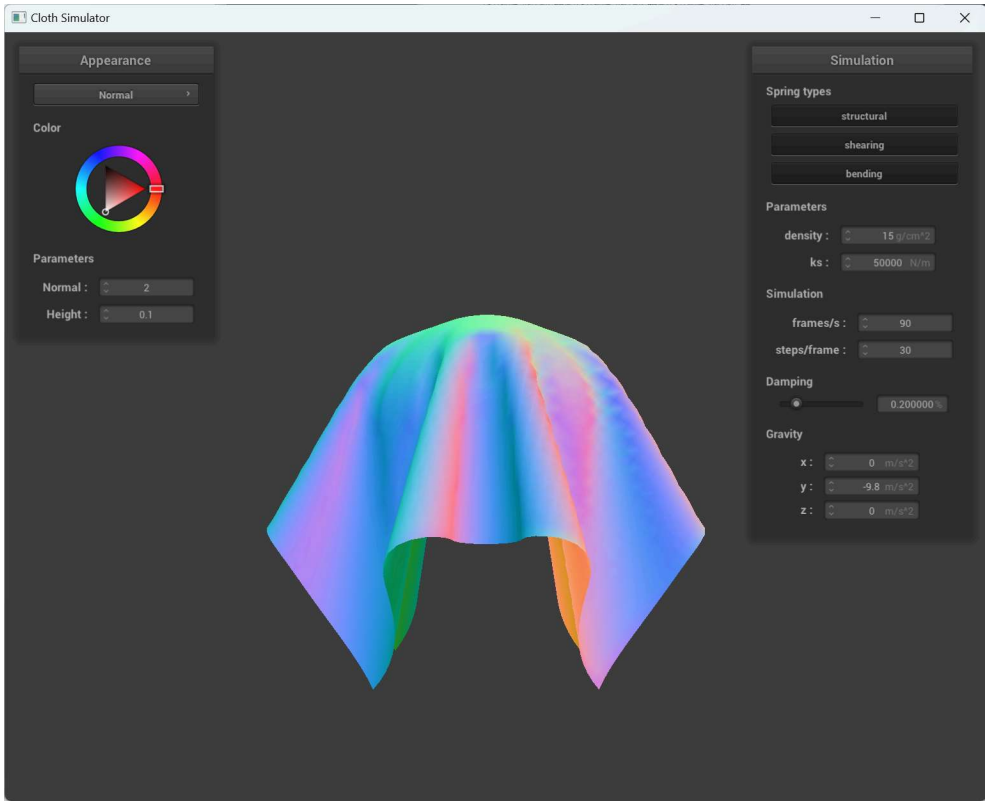
Results from `sphere.json`

Here is a screenshot of the cloth lying on the sphere, with default parameters (`ks` = 5000). Shaded using normal shading.



And the following two screenshots are cloths with `ks` = 500 and `ks` = 50000 respectively.

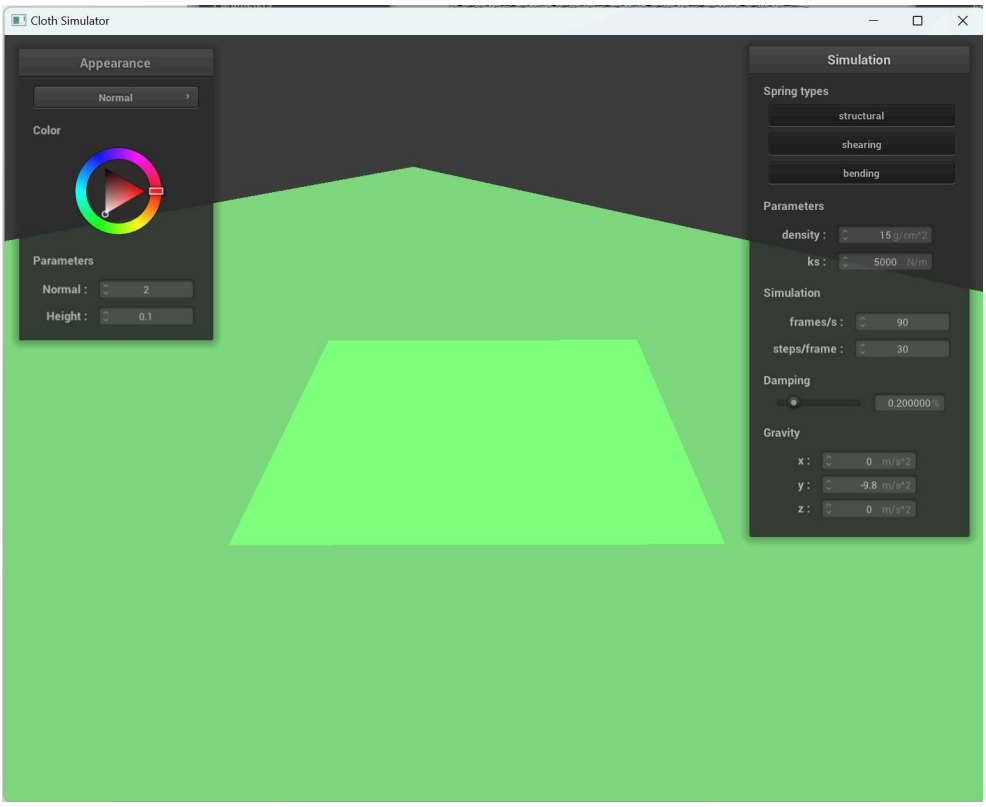




We can see that with a lower spring constant, the cloth is more easily folded, exhibiting more and shallower creases around the sphere. With a high spring constant, the cloth is more bouncy and tends to stick out, and the creases are fewer, wider and deeper.

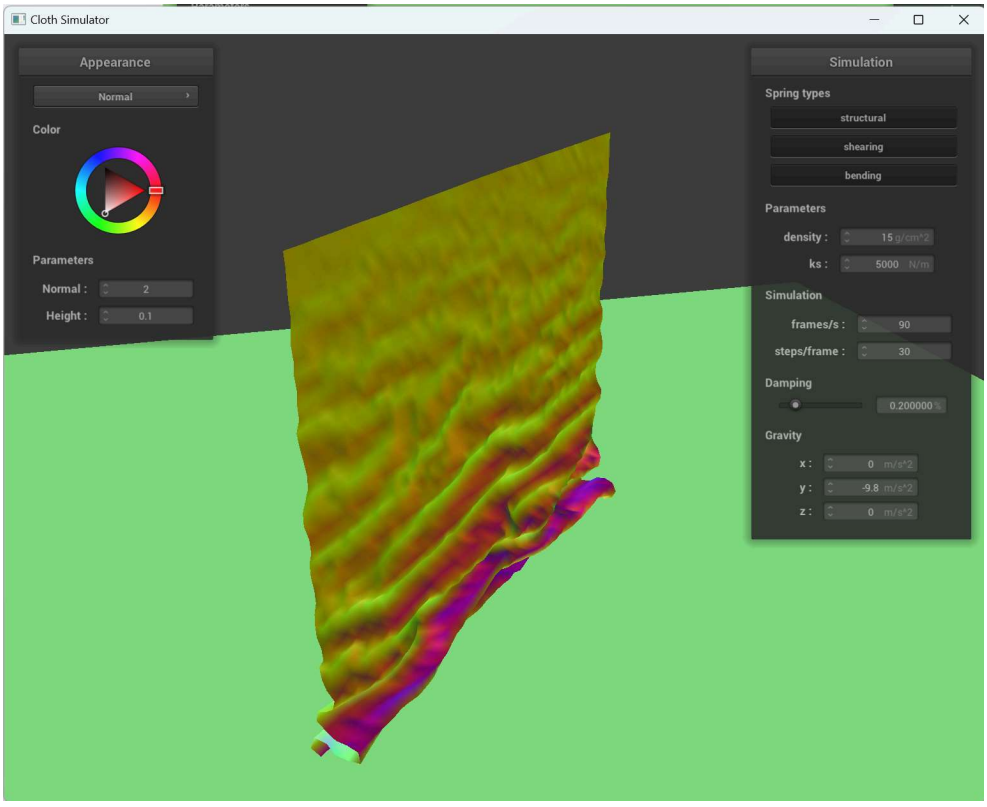
Results from `plane.json`

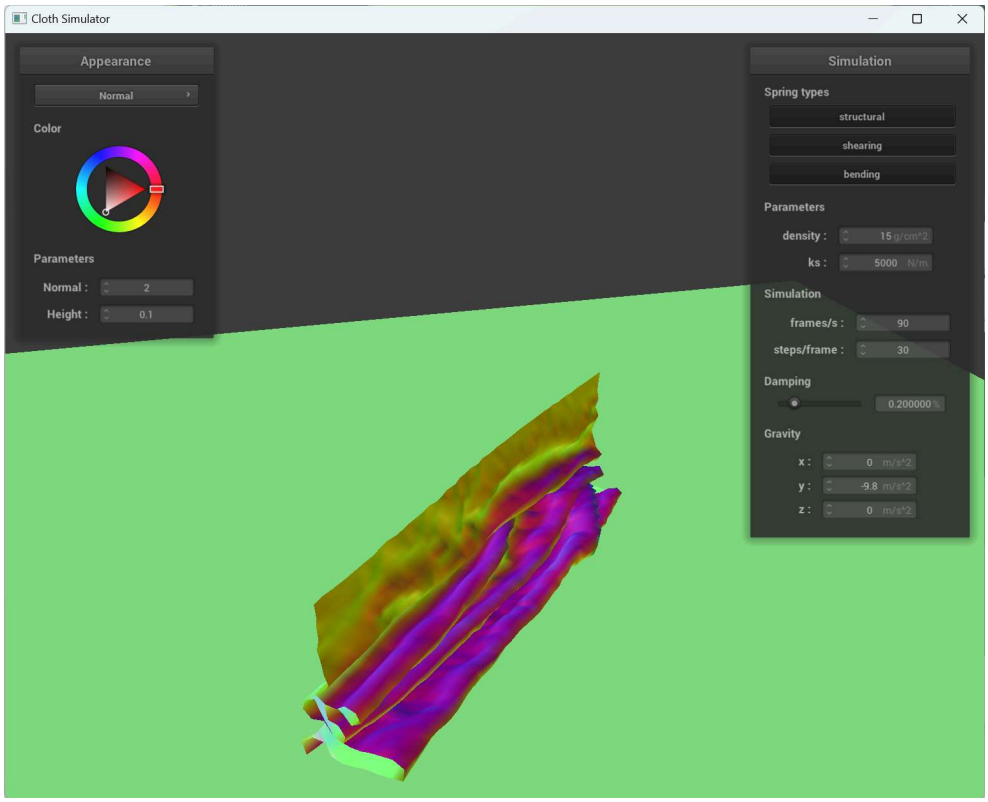
Here is a screenshot of the final resting state of the cloth on the plane, using normal shading.

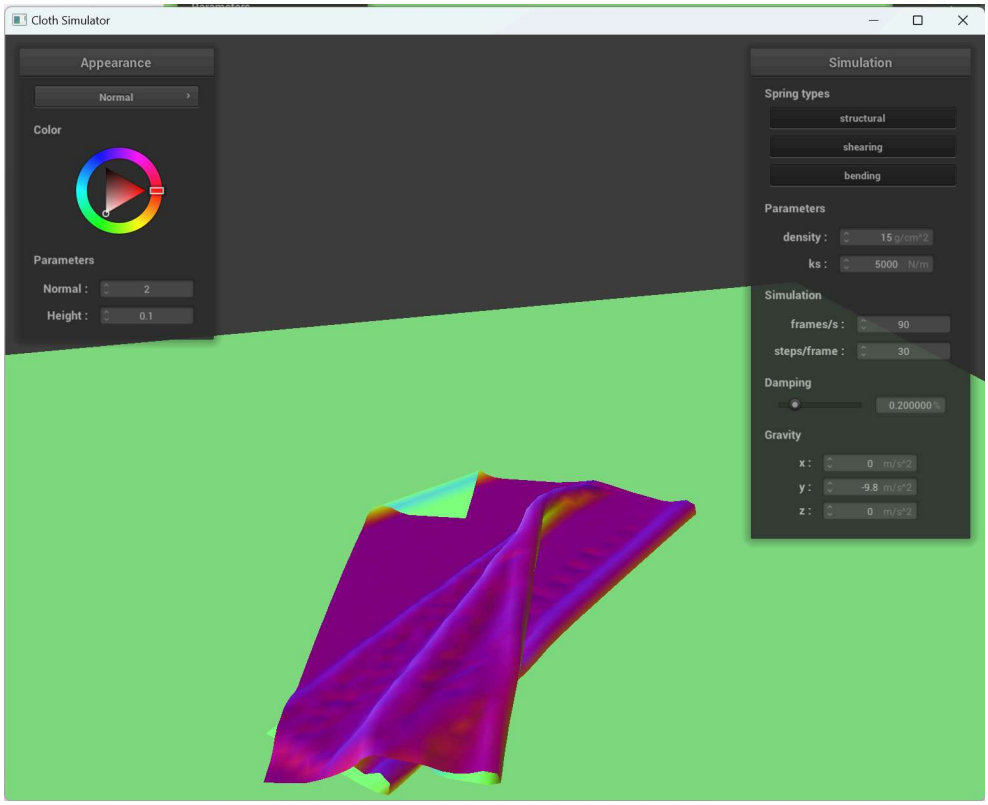


Part 4: Handling self-collisions

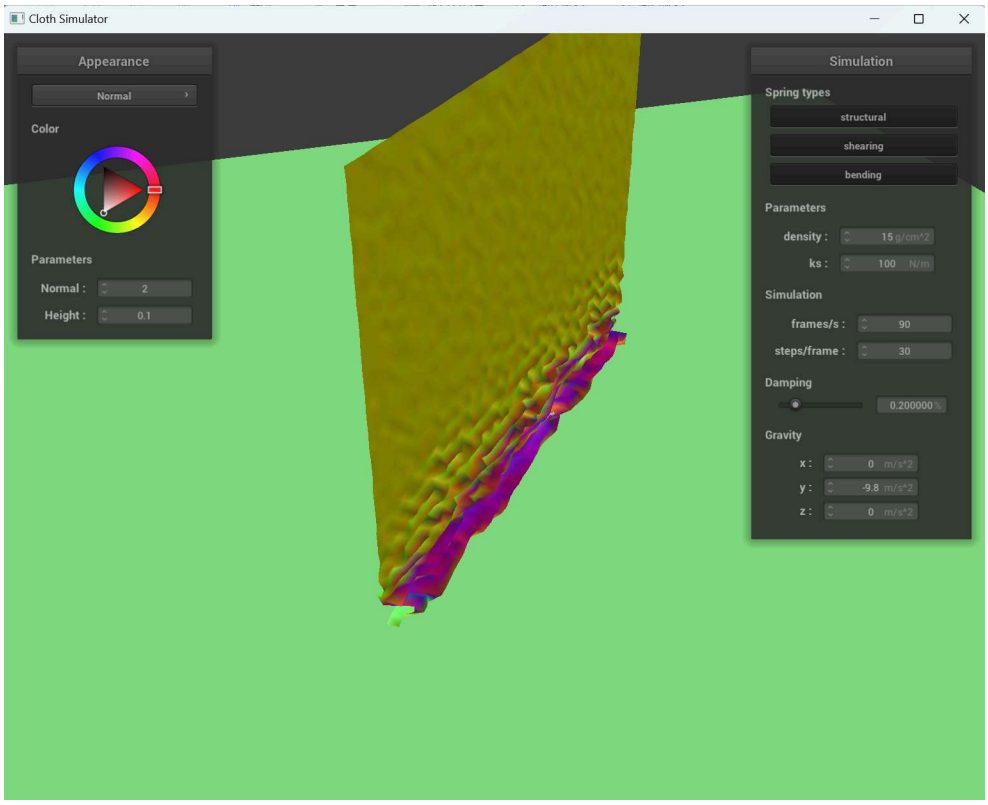
The following screenshots show the falling and folding of a cloth in `selfCollision.json` with default parameters.

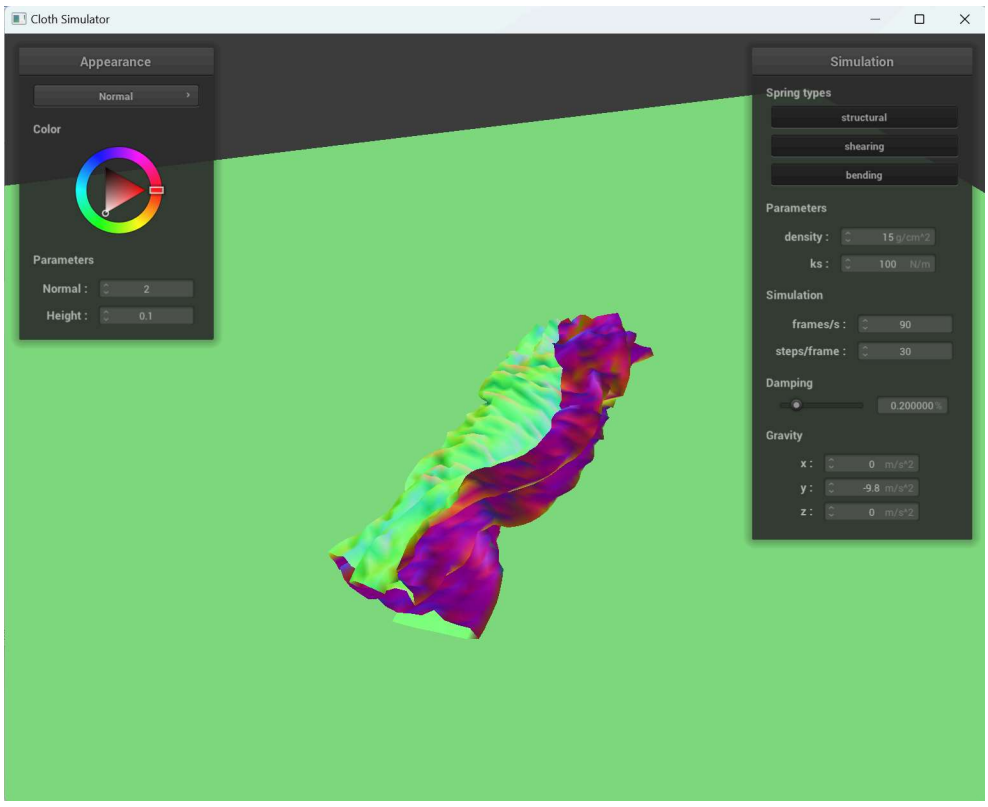




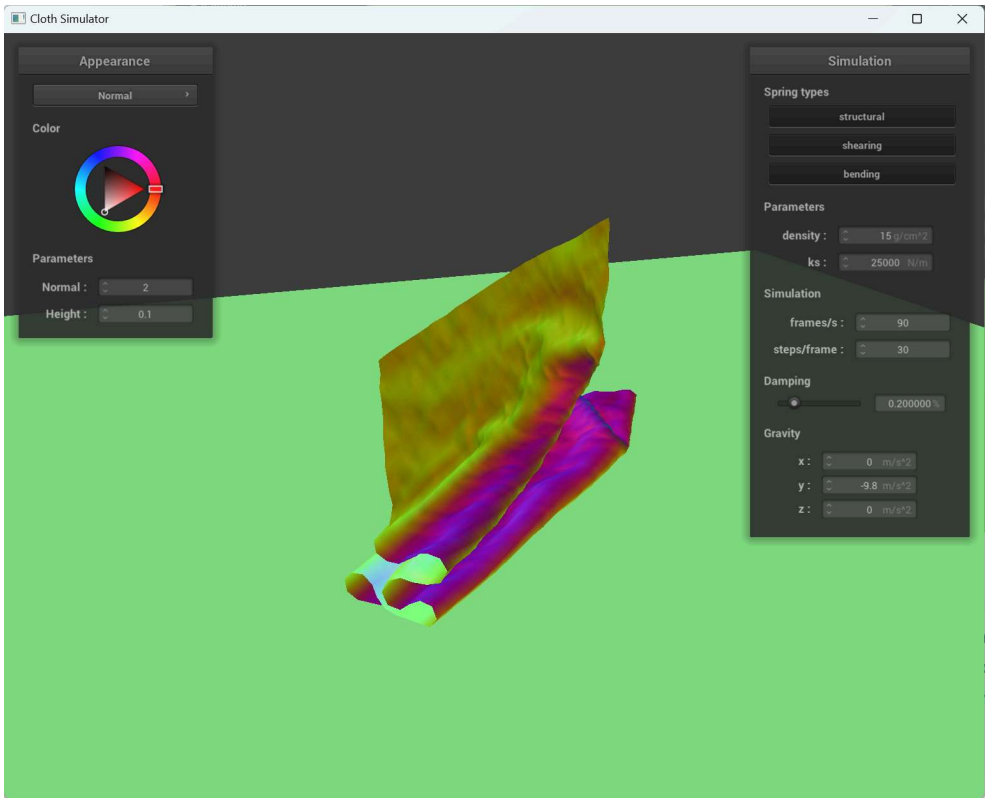


Observations for varying `ks` and `density` are also made. When `ks` is small, a larger number of creases are formed and there are more waves on the cloth as it bounces on the plane, because the point masses move more easily relative to each other. This is what it looks like (`ks` = 100):

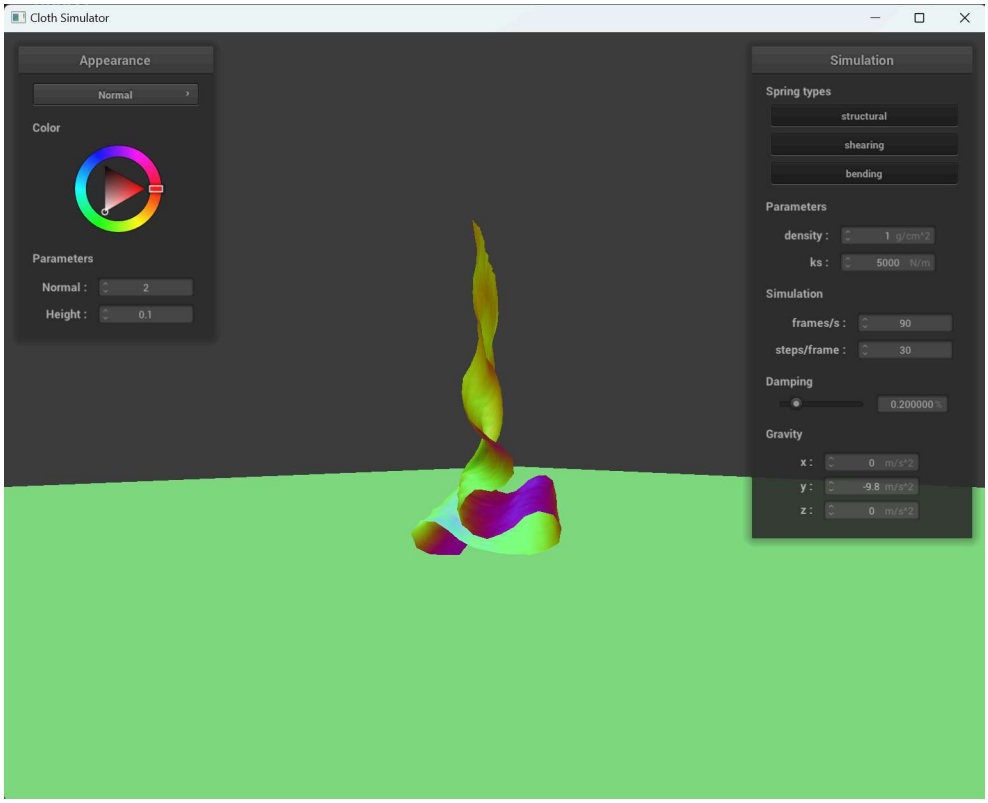




When `ks` is large, the creases are fewer and larger than default, and is more bouncy on the plane: (the following screenshot is taken with `ks` = 25000)



When `density` is small, the internal forces are relatively more dominant in determining the cloth's behavior, and the cloth bends as it falls gently to the plane: (`density` = 1)



When `density` is large, the cloth falls with a flump onto the plane.

Part 5: Shaders

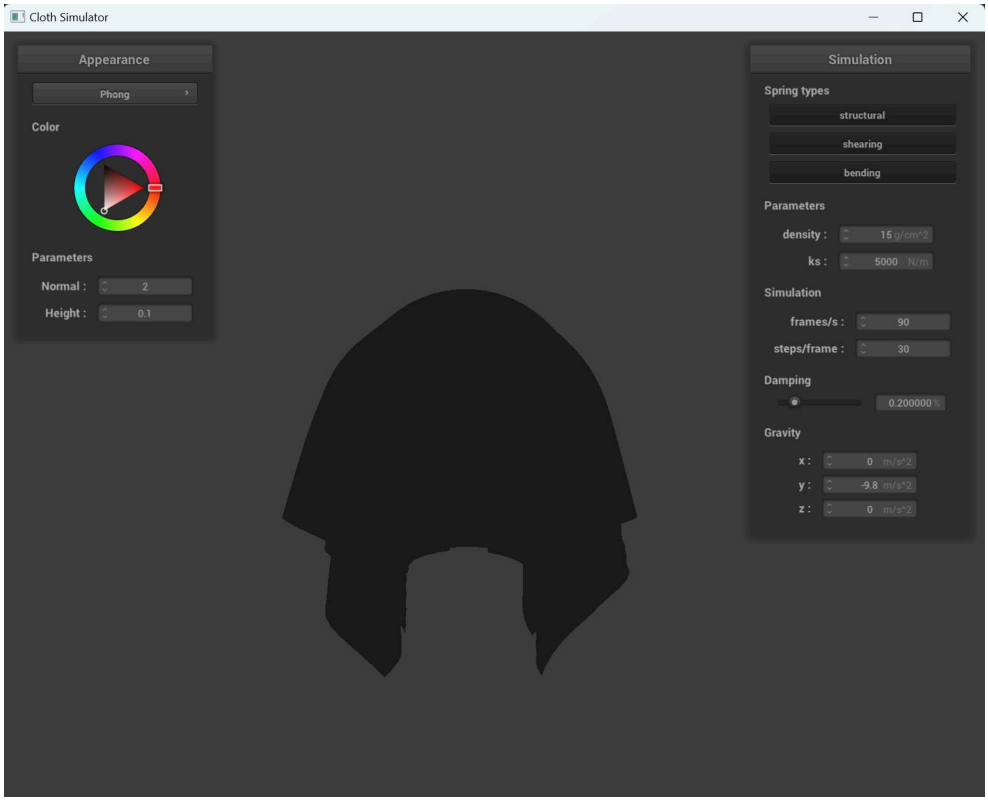
A shader program takes a 3D model as input, calculates how it should look like on the screen, and displays it. Vertex shaders apply transforms to vertices and calculate the vertex positions as well as passing important information about the vertices to the fragment shaders; fragment shaders take the vertex arguments and, according to lighting and shading rules, decide and output the color to be displayed on the pixels corresponding to the vertices. Various shading procedures can achieve various effects of lighting and texture.

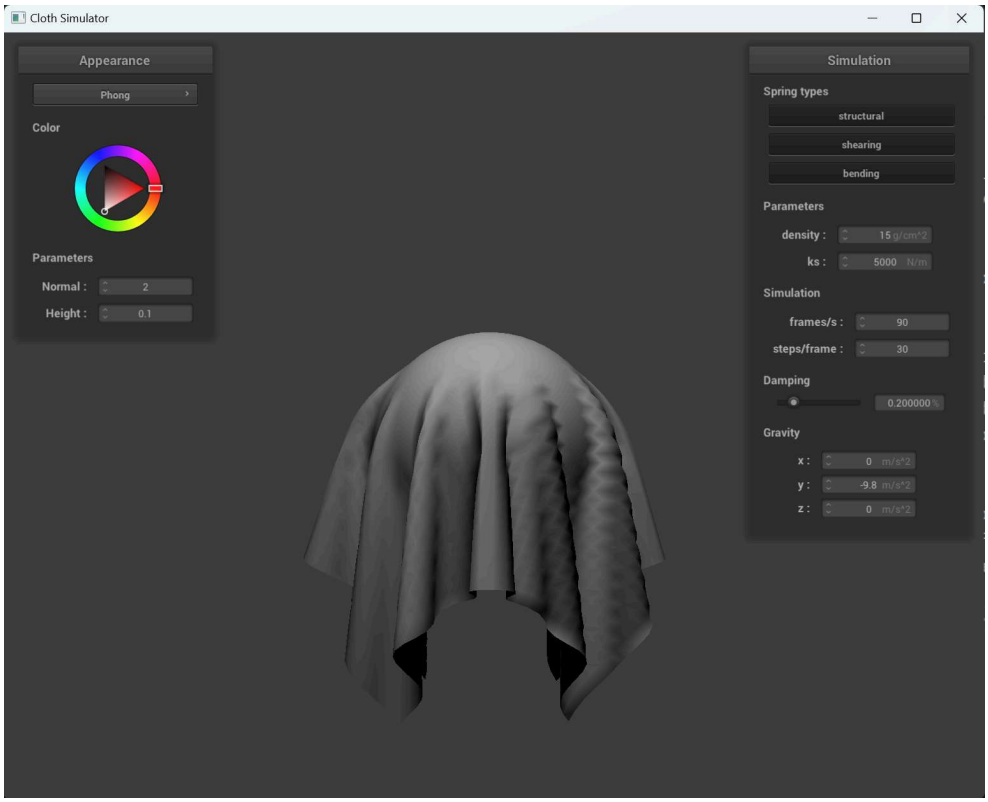
All following results are from the resting state of `sphere.json`.

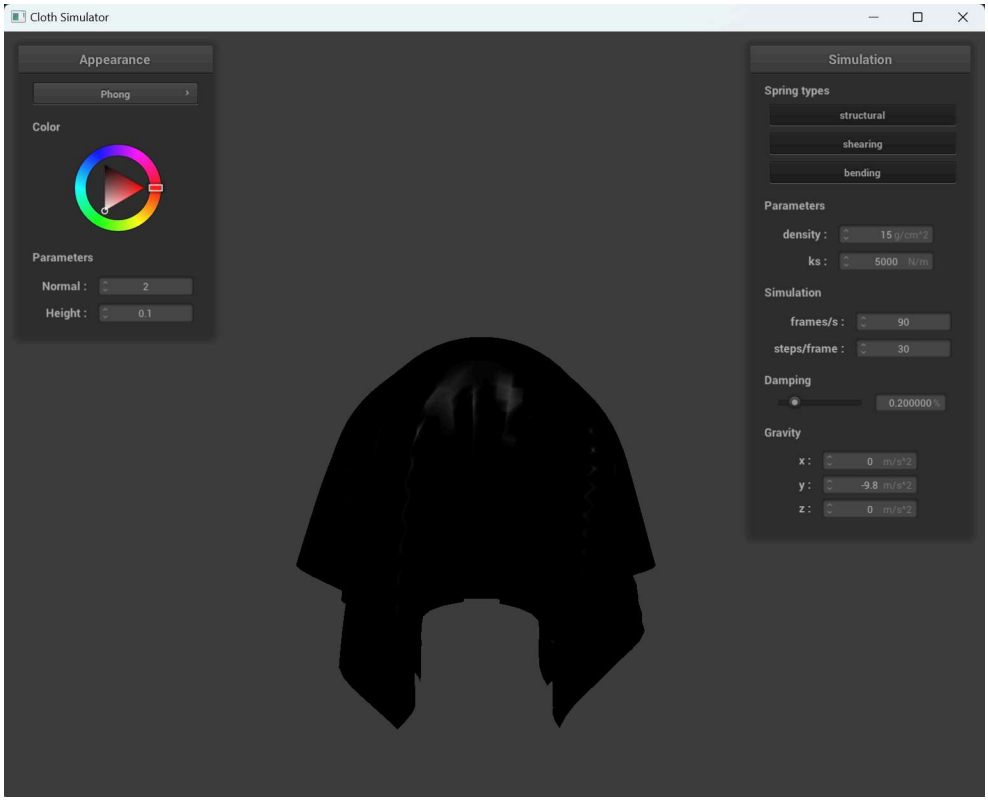
Blinn-Phong Shader

The lighting calculated by the Blinn-Phong shading model has three components: ambient lighting, which serves as background; diffuse lighting, which corresponds with diffuse reflections of light on the material; and specular lighting, which are highlights on the material and corresponds with mirror reflection on the material.

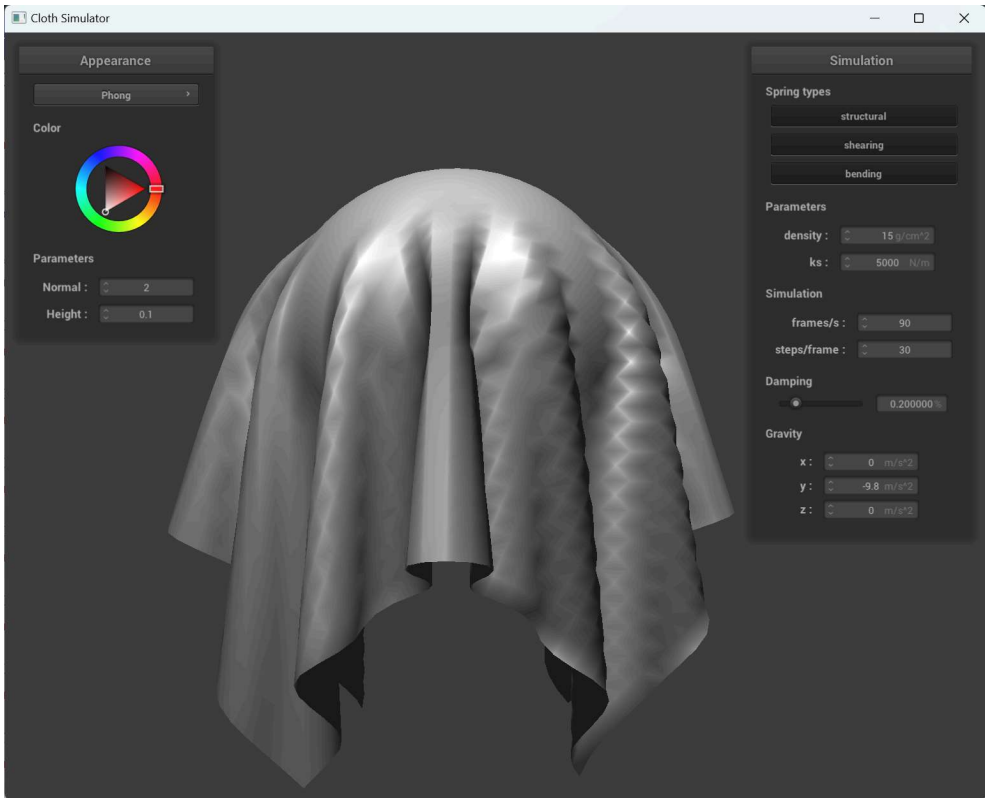
The following are the ambient, diffuse and specular lighting components respectively. Some parameters are as follows: $K_a = 0.1$, $I_a = vec4(1, 1, 1, 0)$, $K_d = 1$, $K_s = 0.5$, $p = 32$.





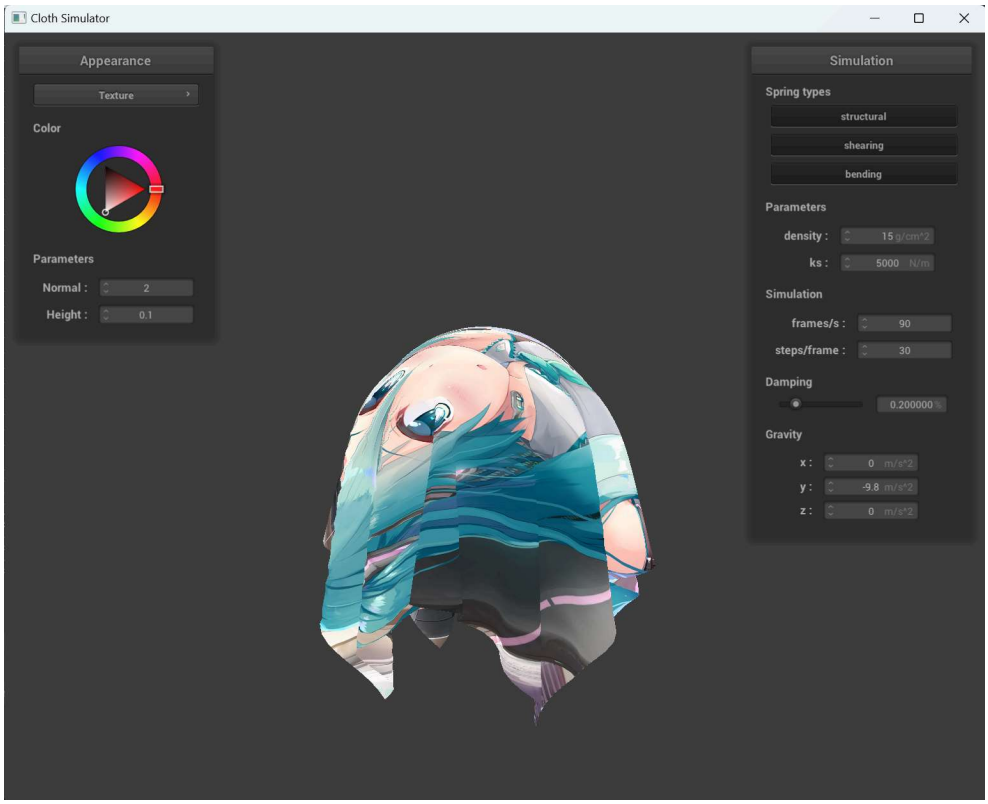


And this is the full Phong shading result:



Texture Shader

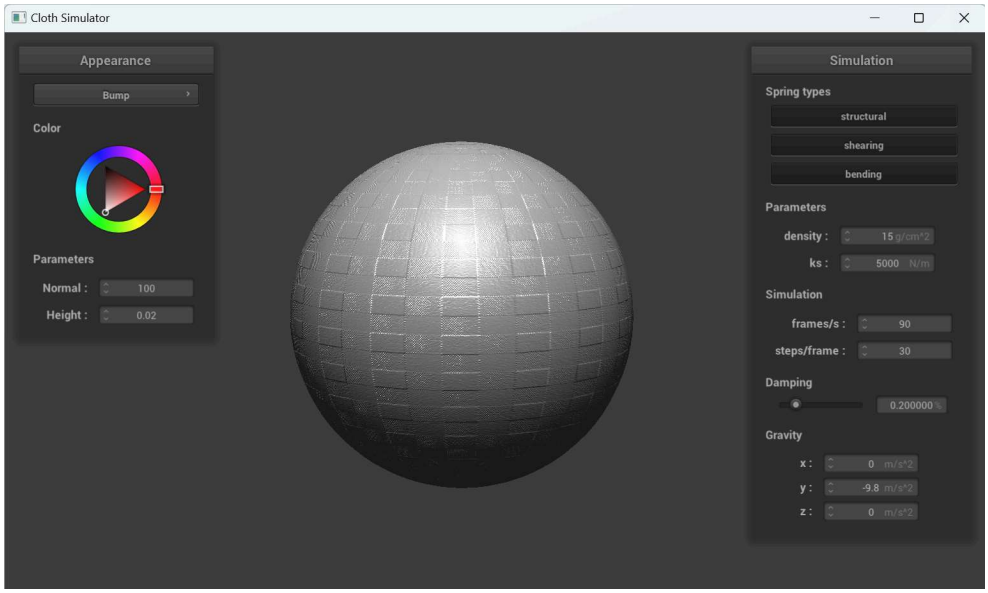
Here is the result of texture shading with a replaced `.png` texture.



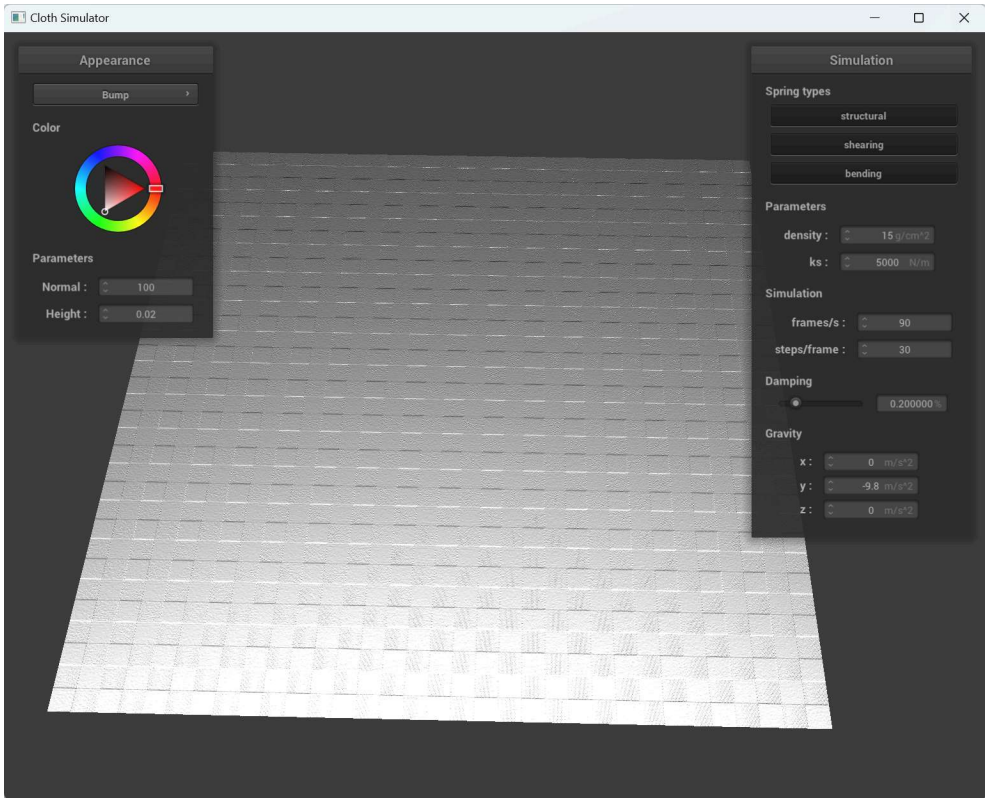
Bump Mapping and Displacement Mapping

`texture_4` is used for bump and displacement mapping, and some other parameters are Normal = 100, Height = 0.02.

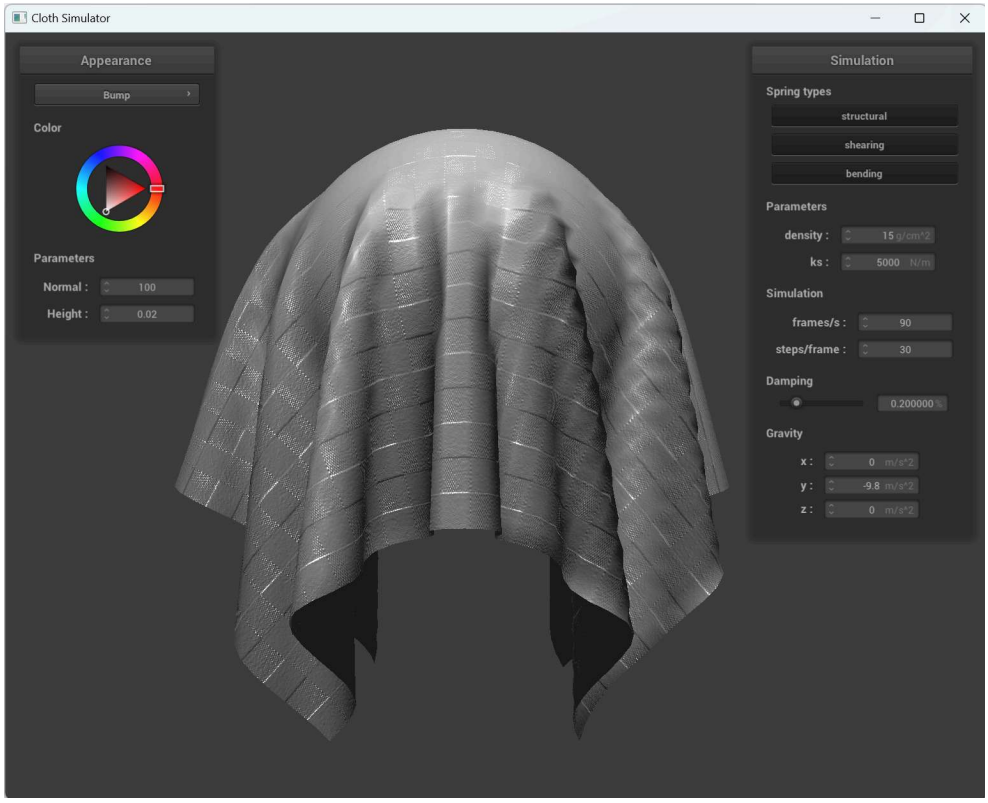
Here is the sphere rendered using bump mapping with default parameters excluding above:



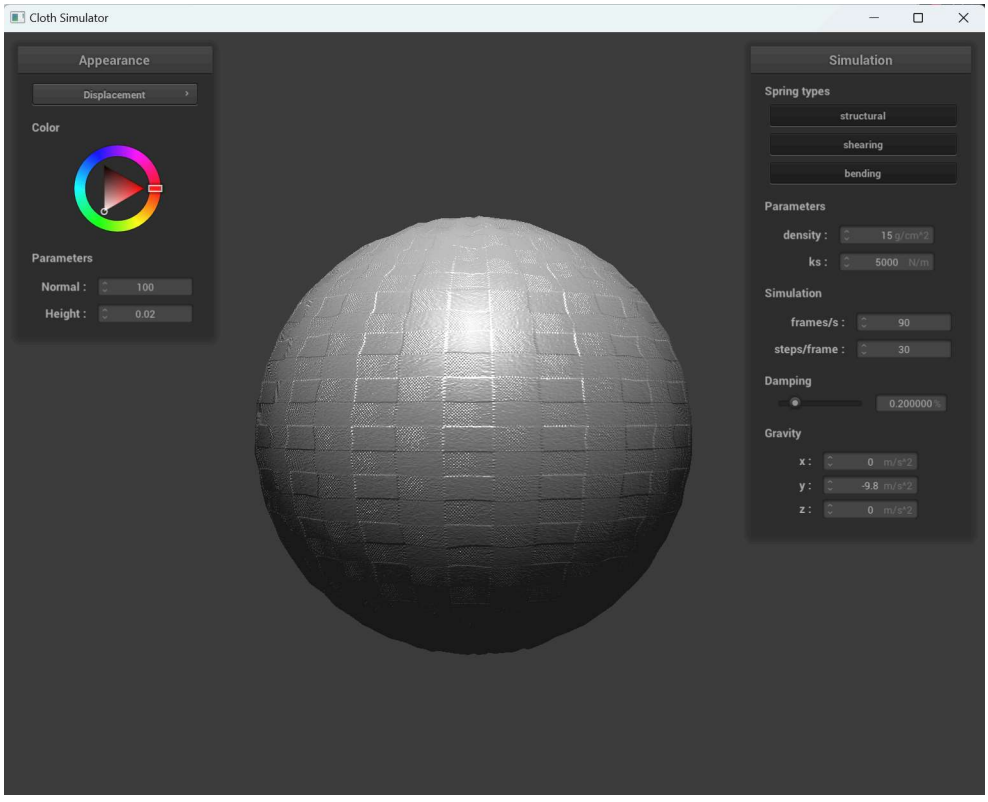
Here is the cloth in its starting position:



And the cloth resting on the ball:

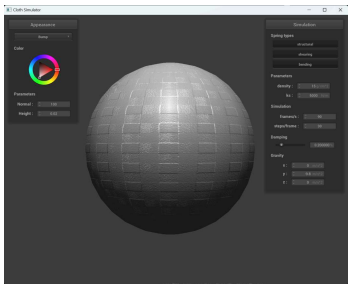
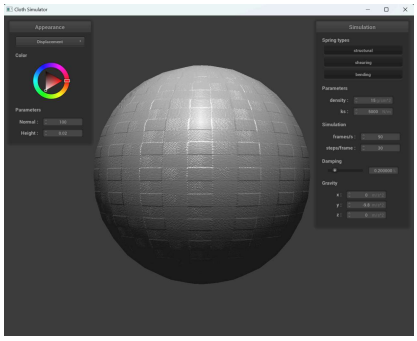
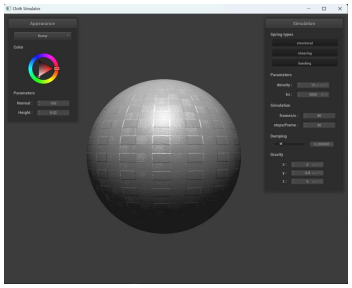
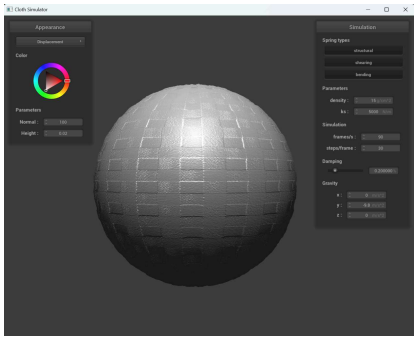


This is the sphere shaded using displacement mapping instead:



Compared to the same sphere shaded using bump mapping, this sphere has real bumps on it, corresponding to the regions of high and low values of the height map; the previous one was very smooth in terms of vertex positions.

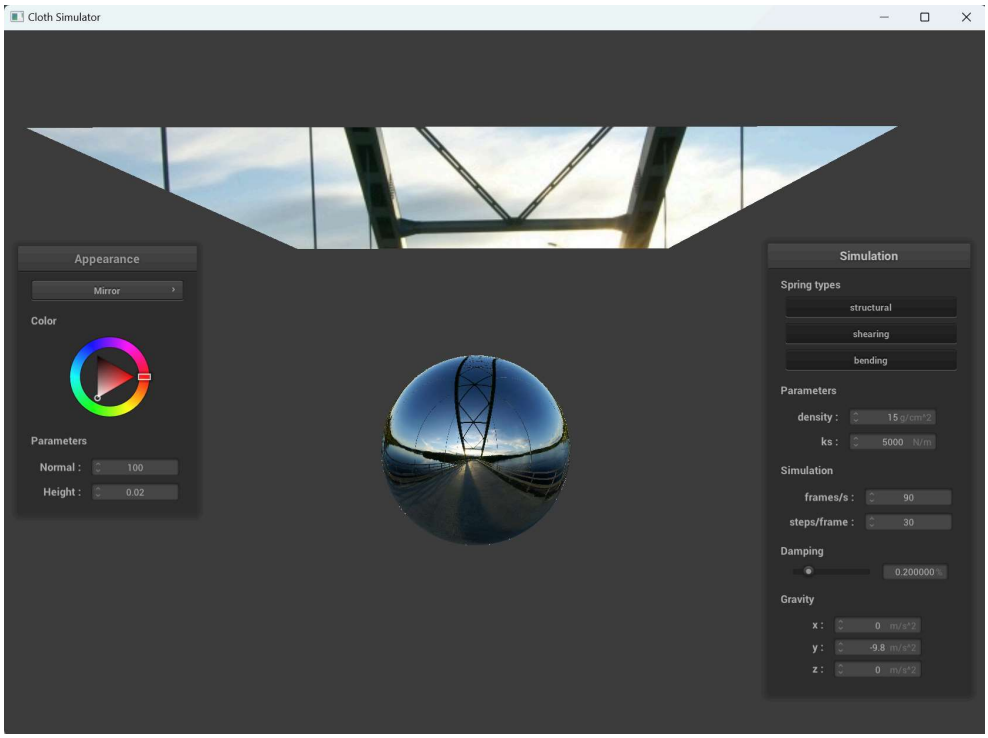
The following are results of bump mapping and displacement mapping on the sphere with varying mesh density of the sphere.

	Bump Mapping	Displacement Mapping
`-a 16 -o 16`		
`-a 128 -o 128`		

In each row, the camera is placed at the exact same position. We can see that displacement mapping noticeably inflates up the sphere compared to bump mapping. On the sphere with coarser mesh, the bumps produced by displacement mapping are less obvious, because the texture has a recurrent pattern and a coarse mesh averages out the bumps. On the sphere with fine mesh, the bumps are more visible and fitted to the texture pattern.

Mirror Shader

The result of the cloth AND the sphere shaded using mirror shader:



The result of the cloth resting on sphere: