Strawberry Ice Cream

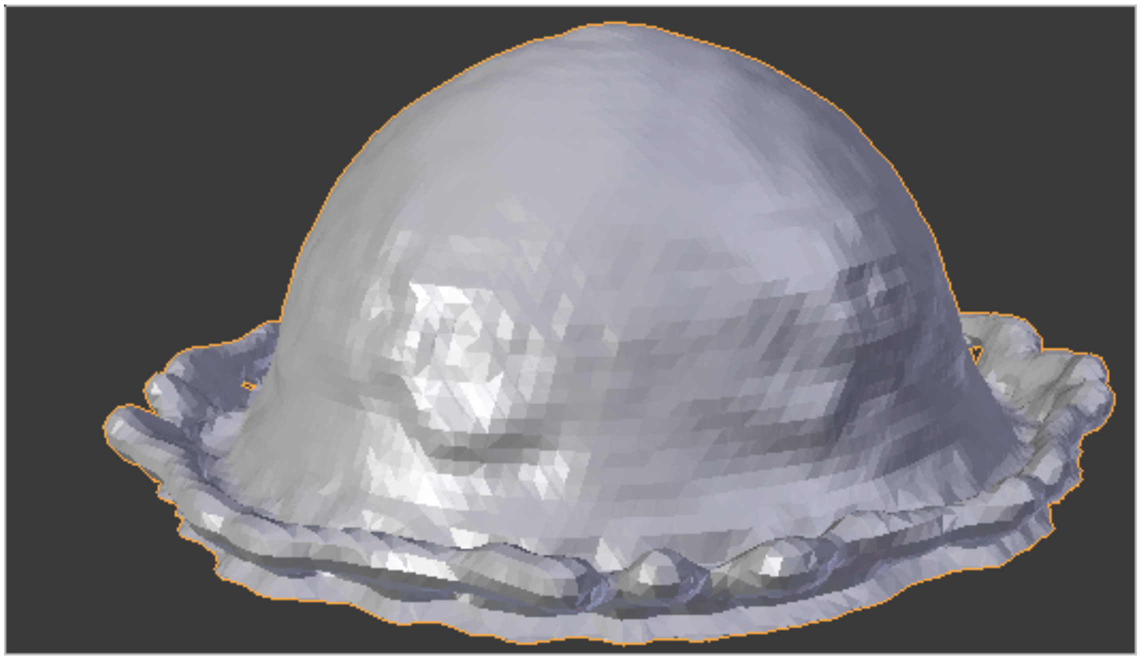
Brennan Shacklett and Peter Do

Modeling the Ice Cream:

Our first approach was to simply attempt sculpting the ice cream from a sphere manually in blender, but we rapidly discovered that our sculpting skills and the tools available in blender were simply not up to the task. Ice cream has a significant amount of complex ridges, overhangs, and irregular geometry that are all difficult to sculpt with such fine detail. Blender's sculpt tools seem to work poorly without a fine enough mesh, but the level of detail we desired required such a fine mesh that everything slowed to a crawl and it was impossible to make progress.

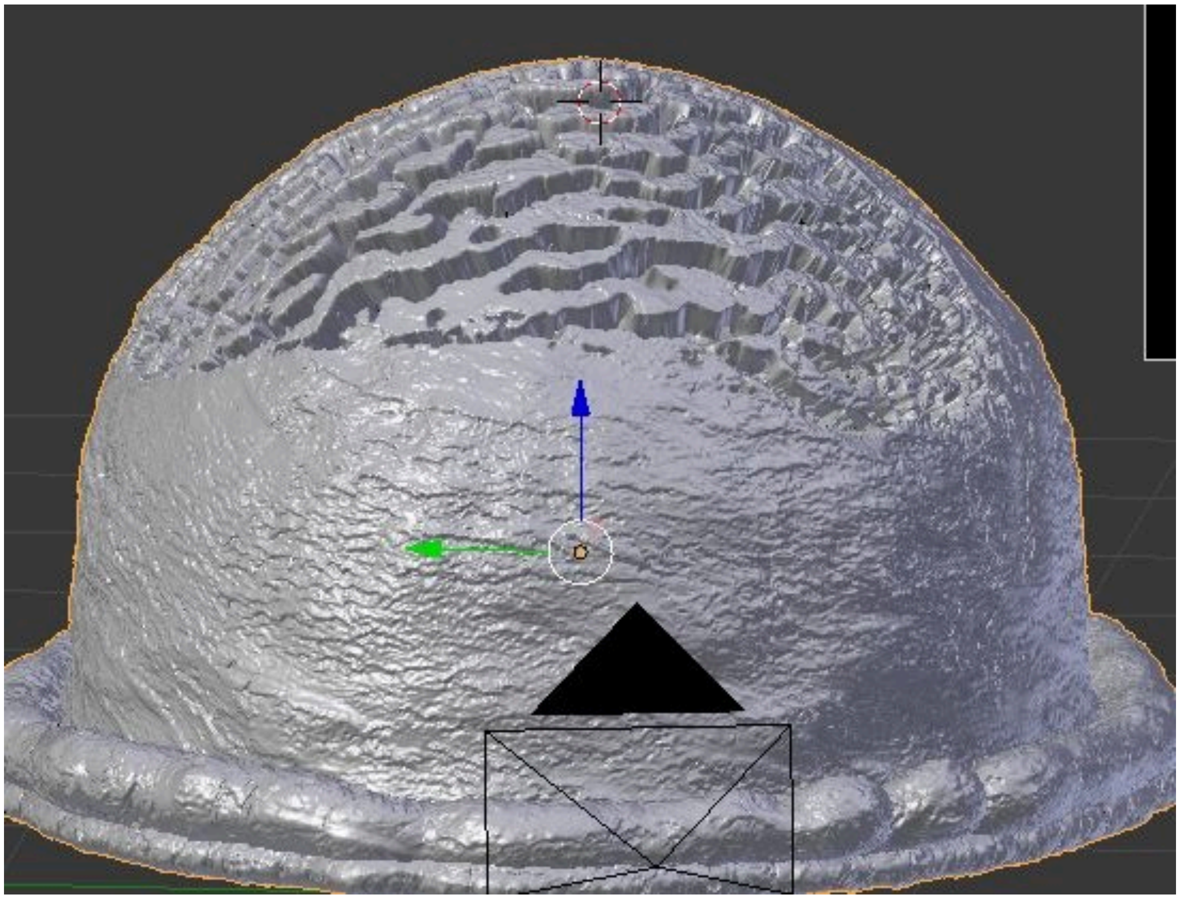
We initially decided that creating a model of *fresh* ice cream was simply infeasible due to the number of vertices required, so we decided to try and model ice cream that had just started to melt instead. To begin, we ran a fluid simulation of an extremely thick viscous ball of liquid dropping down onto the ground. The frame where the ball impacted the ground created a very nice base model for the melty ice cream.

Fluid Simulated Base Model



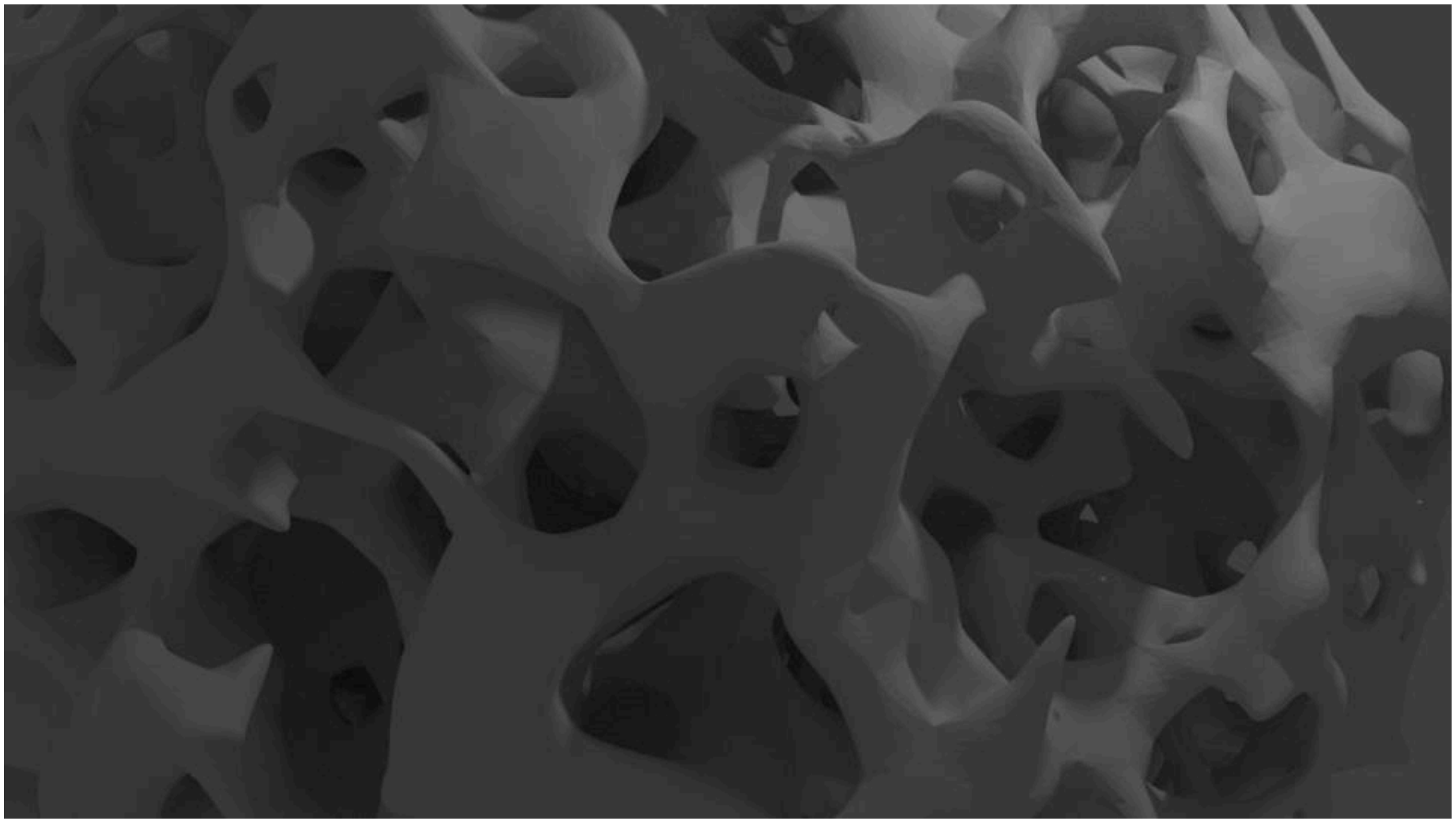
We still had to add detail to our base however, and the first natural idea was to displacement map existing ice cream images onto our base model. Unfortunately, this technique did not provide us with realistic results. Displacement mapping along normals can cut out grooves, but it can't provide any of the overhangs or interesting geometry within these grooves. Ultimately we were unable to come up with a good technique for modelling the deeper, more complex features of ice cream, since most attempts simply looked too regular and unnatural.

Displacement Mapping



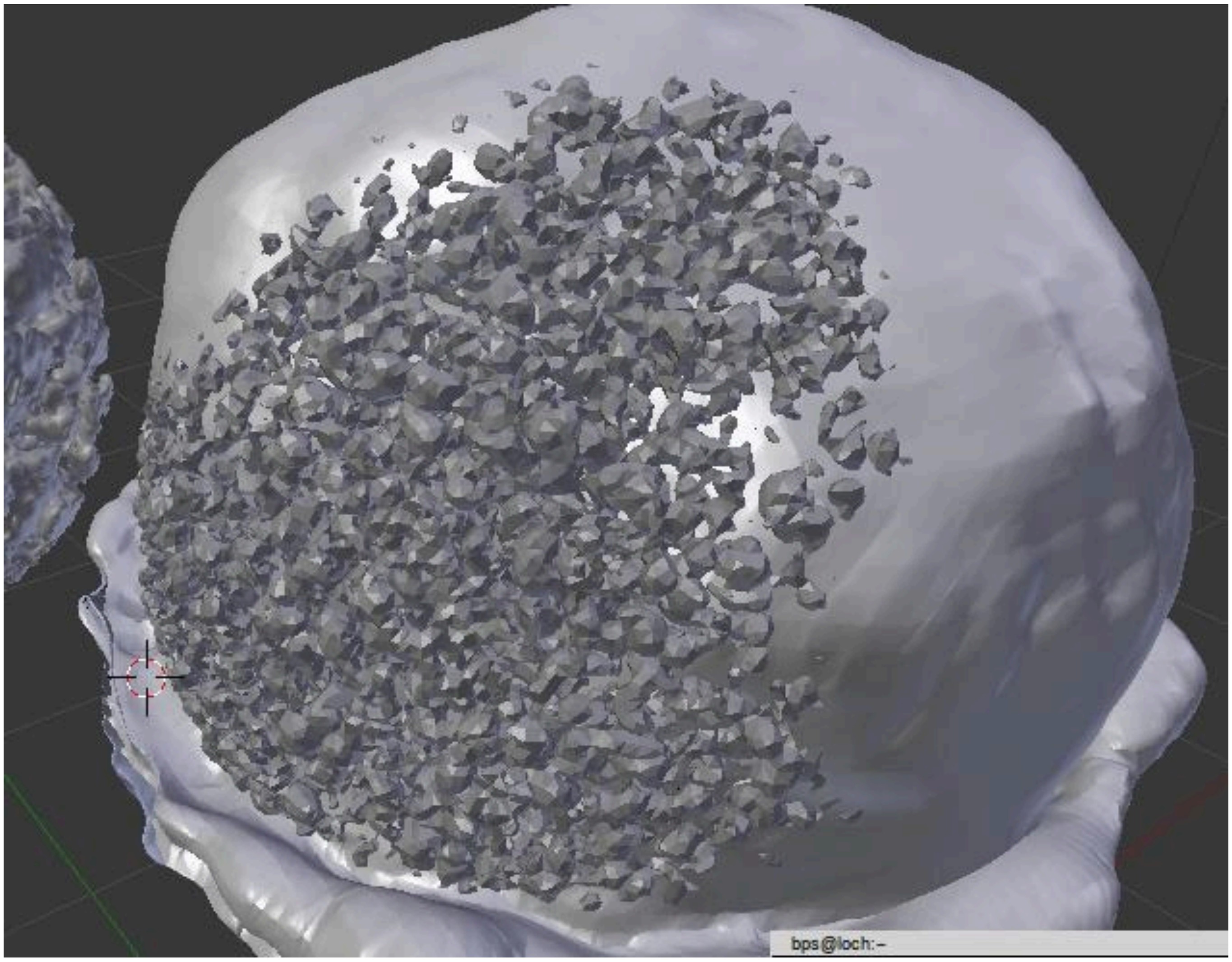
We briefly considered using vector displacement (different colors of the displacement texture would displace in different directions), but without dedicated tools it isn't possible to get coherent results.

The final technique we settled on was inspired by the 2014 French Bread entry: using random noise to create a point cloud that was converted into a mesh with marching cubes. We noticed that the inside of the bread had a similar basic structure to what we desired for our ice cream, although the icecream needed to have larger variations. We modified an existing and outdated plugin for blender that performed marching cubes on blender particle systems so it could instead read a randomly generated point cloud from disk. Then we implemented variations on voronoi and opensimplex noise to try and get the results we wanted. The first implementation we tried was simply using the values returned from the noise function for a given (x, y, z) point as a density value, and then we meshed points above a certain isolevel and dropped points below that level. This resulted in stringy looking random meshes that were not remotely ice cream, shown here:



We realized the problem was that it was very hard to recreate the basic shape of ice cream from these random meshes, since we didn't have a good way of specifying that large portions of the mesh should be simple and connected (everything looked stringy). We also tried making the noise mesh less stringy using displacement modifiers but combining this with the base icecream mesh didn't look convincing.

Using Displacement Modifiers



We solved this problem by using blender boolean operators to effectively use these random noise meshes to cut parts out of the base icecream mesh, which generated the level of detail we required.

Modeling other items:

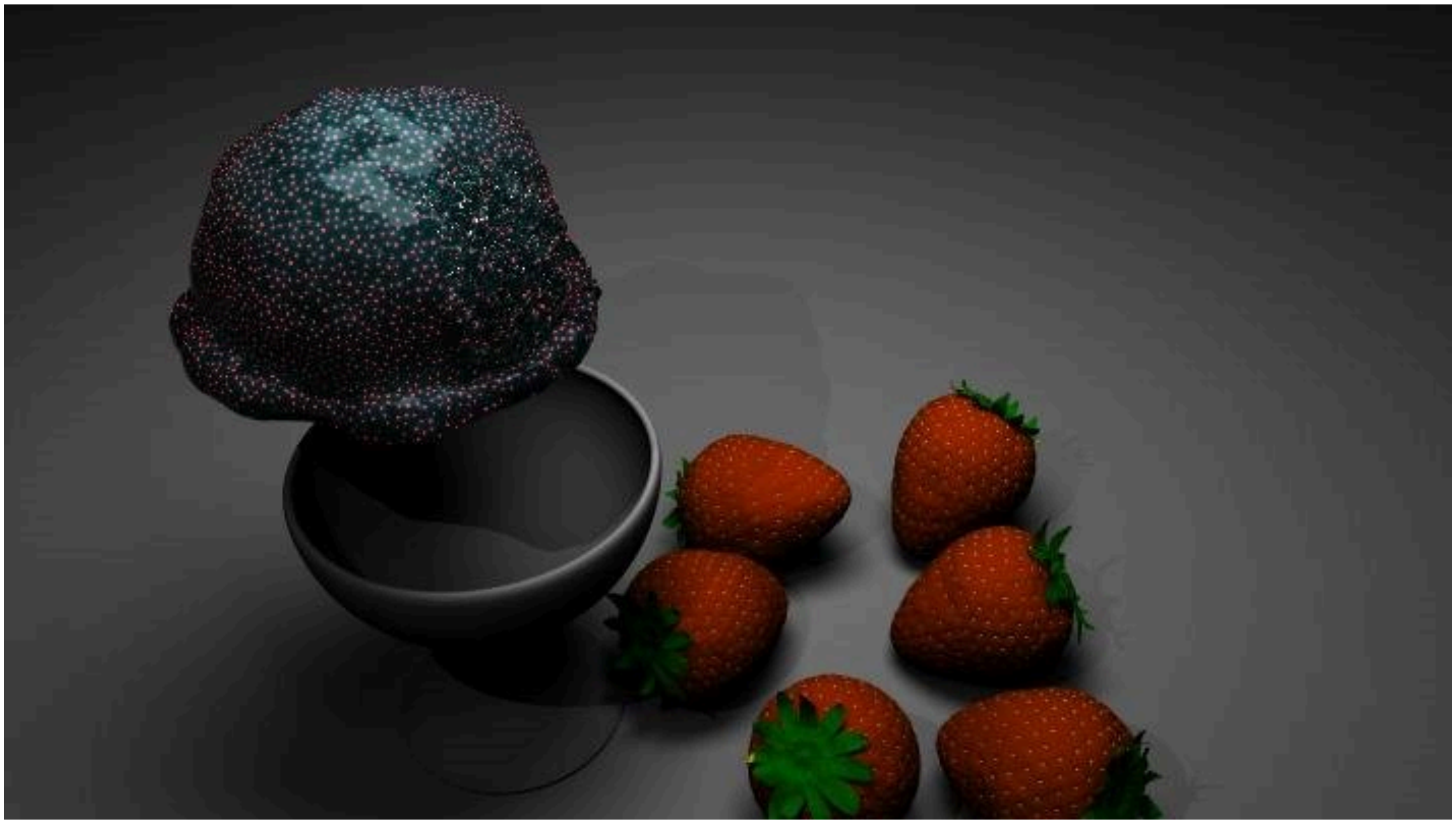
Strawberries
We began by modeling strawberries ourselves in blender. The results were decent and could have been used in our final image. However, we found some nice strawberry models online and incorporated those for our final image.

Cup
The cup went through a couple of iterations as we wanted to make it look like a realistic sundae cup. We first modeled a round bowl shape, but found that it was a little too wide. Deepening it made it look more like a wine glass. After studying sundae cups we decided that using blender?s capabilities to mold a cube would be better than starting with a sphere, and allowed for the *flare* that the cup has now. Sundae cups often also have

angular shapes and aren't perfectly round, so we thought that the flare and beginning with harder edges would carry that through to the final model, and it did. All of the modeling of the cup was done in blender.

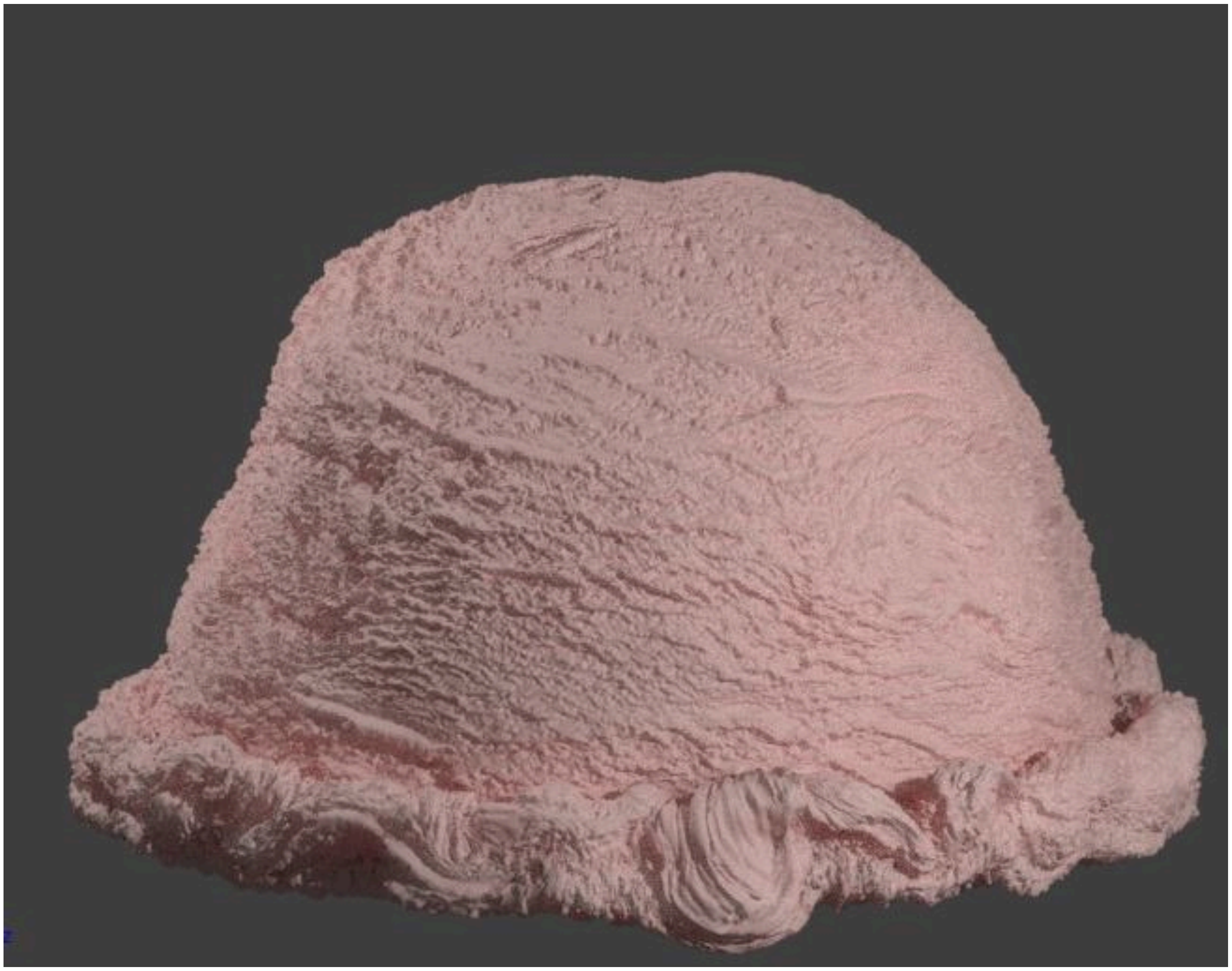
Rendering the Ice Cream:

Our original plan was to modify PBRT's built-in subsurface scattering capabilities to create a more realistic subsurface scattering effect for the ice cream (and objects in general). We first tried using the skin model outlined in http://graphics.ucsd.edu/papers/egsr2006skin/egsr2006skin.pdf with PBRT's dipole approximation, but rapidly ran into trouble finding parameters that looked decent for icecream. We decided to step back and try to find the sigma*a and sigma*prime_s that worked with PBRT?s built in implementation (in order to rule out major bugs in our implementation), but we ran into more serious issues. It seems that the dipole approximation has a serious bias towards towards blue green scattering which makes creating something like pink ice cream incredibly difficult. Using the built-in kdsubsurface material with a pink Kd term and low mean free path resulted in a blueish green jello like substance covered in pink dots.



We tried switching to manually inputting sigma values into the subsurface material, but there was a clear trend: either the material was blueish and extremely dark, or it was covered in bright dots. We managed to mitigate this problem somewhat by decreasing the minsampledistance parameter on the dipolesurfaceintegrator, but this seemed to exponentially increase the rendering time and memory requirements to the point where it was infeasible. Even with normal settings, the dipole integrator increased render time by at least 3x, which made using trial and error to find the correct parameters essentially unworkable. We had already managed to create a fairly pleasing icecream material in blender by mixing blender's sss material and a diffuse material, so we decided to implement blender?s sss in pbrt.

Ice Cream in Blender



Since blender?s render cycles uses a path integrator, we worked in blender's bssrdf implementation into pbrt's path integrator, based on blender's code base and the papers referenced here: https://wiki.blender.org/index.php/Dev:Source/Render/Cycles/Subsurface_Scattering. Our implementation is somewhat simplified, and due to the fairly different architecture between cycles and pbrt, the parameters unfortunately do not carry over from blender correctly. Nevertheless this gave us a more functional version of sss for the purposes of the ice cream than the dipole approx in pbrt, so this is what we used for the final image.

Rendering other items:

As the other items didn?t require complex subsurface scattering, we primarily utilized PBRT's built-in materials. This required extensive trial and error as we wanted to get the most realistic results out of all of the objects in the scene, from the glass cup to the strawberries. For the cup, we decided the default *glass* material would work. To accurately render the strawberries, we first attempted to mix the modified PBRT subsurface material with a plastic material. The results were pretty strong using the dipolesurfaceintegrator, but after moving to path integration, we reconstructed our results using modified parameters on the *plastic* material. To make texturing the whole strawberry easier, we had to separate the model into seeds, strawberry body, and leaves. We then textured them all separately, and the seeds used the *uber* material. To texture the leaves appropriately, we used the *translucent* material utilizing an image texture that we tiled over the whole object. The ground was also textured in the same way, but using the *uber* material. The backdrop was simply a textured plane using an image. Overall, rendering the miscellaneous materials was an exercise in trial and error primarily, but getting the separation of objects, textures, and materials just right took a while.

Team contributions:

Brennan: Ice cream noise modelling, implementing blender sss, lighting
Peter: Scene composition, sss parameters, glass cup, ice cream liquid, strawberries, lighting, fluid simulations, backdrop, and ground

Final Result

Resources

Donner et al's empirical BSSRDF model: http://www.cs.berkeley.edu/~ravir/empbssrdf.pdf
Donner and Jensen's skin model: http://graphics.ucsd.edu/papers/egsr2006skin/egsr2006skin.pdf