Team Members : Raymond Paseman, Vincent Huynh
Project : Project 7 - Final Project
Title : The Lonely Warren Bear, The Warren Bear is Alone for Christmas
(The Warren Bear Doesn't Want to be Alone for Christmas)

Theme :

Around this time of year, every year, the students of UCSD depart from the campus to escape from the painful confinements of education. They return to their friends and families to reunite, converse, and feast in celebration of Christmas and New Years. Their minds wander away from reality as they engage in boisterous activities. Everyone is at liberty from their personal dilemmas and without a care in the world.

However, back at the UCSD campus, one poor lonely soul has been left behind, all alone in the empty silence. From the day the Warren Bear was born, he has experienced loneliness in the limited confines of the engineering quad lawn of the Warren College Campus. Upon first sight and joy of the holiday season, the students have temporarily forgotten and abandoned the Warren Bear. All alone, Warren Bear cries himself a river anxiously awaiting for the day that the celebrations end and the students return to him. He cries for attentions in hope that he will find some companionship to get him through these cold, tough times.

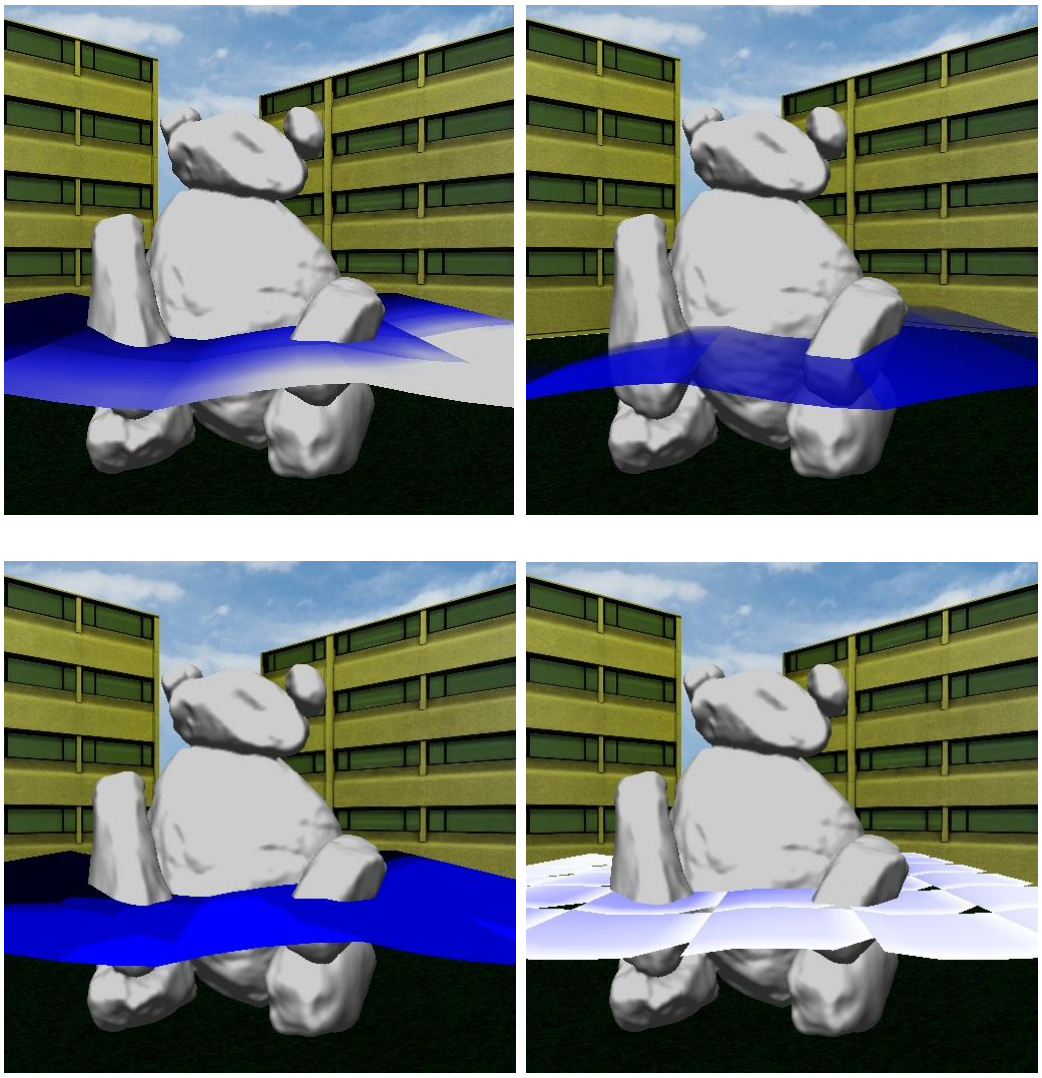
Technical Features :

- Particle Systems (Particles and Particle Engine was used for crying tears)
- Toon Shading (can be run through the Warren bear and water pool of accumulated tears)
- Water Effects (
  - shader for Fresnel Effect with light
  - animated small waves with changing generated Height Map
  - transparency through alpha blending
  - (possible other small details such as ripples or minor turbulences)

)
- Bezier Patches (alternative way to display the water and waves) [incomplete]

Techniques :

Our first endeavors began with creating the focus of our project, the graphical representation of water. We began at first by using a grid of many Bezier patches to portray the surface of a body of water. The Bezier patches would have changing control points to animate some moving small waves within the water. However, we ran into some complications dealing with the C1 continuity within a grid of Bezier patches. It was simple to deal with C1 continuity with a 1-dimensional/direction attachment of Bezier patches (as in a 2-patch piece flag), but it was slightly more troublesome with 2-dimensional/direction attachments of many patches to create a grid collection of Bezier patches representing our pool of water. This ultimately lead to issues calculating the proper normal vectors at the joining edges. Normal vectors are the determining factor to produce proper effects with the light interaction properties of water.



(picture of buggy Bezier patch water with multiple shaders)
(no shading, point-per-pixel, toon, fresnel)
(note: the size of the Bezier Patch version of the water didn't span beyond the limited view)

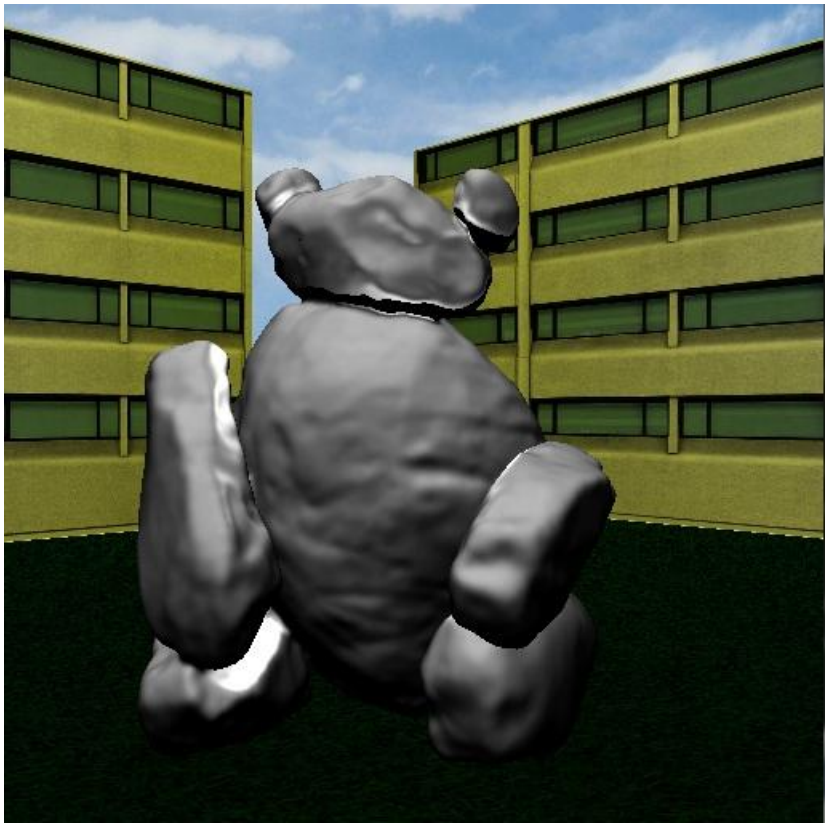
Assuming the Bezier patches were fully functional, it would suffice for our body of water. At this point we decided to try out an alternative method to represent water, using height maps to generate our body of water. The waves and animations of the water are now produced by creating a height map using the product of a sin function and cos function using the u-coordinates and v-coordinates along with an increasing time value to vary the waves in different locations.

(picture with Height Map generated water)

Our next focus was to add the bear in as the centerpiece to our scene. The Warren Bear was provided to us through an .obj file containing the vertices to draw the bear onto the screen. It was a simple task. We produced two shaders to show the Bear in a different way. One was using a more natural method to color to bear in the given lighting. It was a shader to shade the Bear for a point-light

with Per-Pixel Shading using the Phong Illumination Model. We then produced a Toon Shader to shade the Bear in a less serious and fun manner. It provides a funny cartoon way to display the scene.

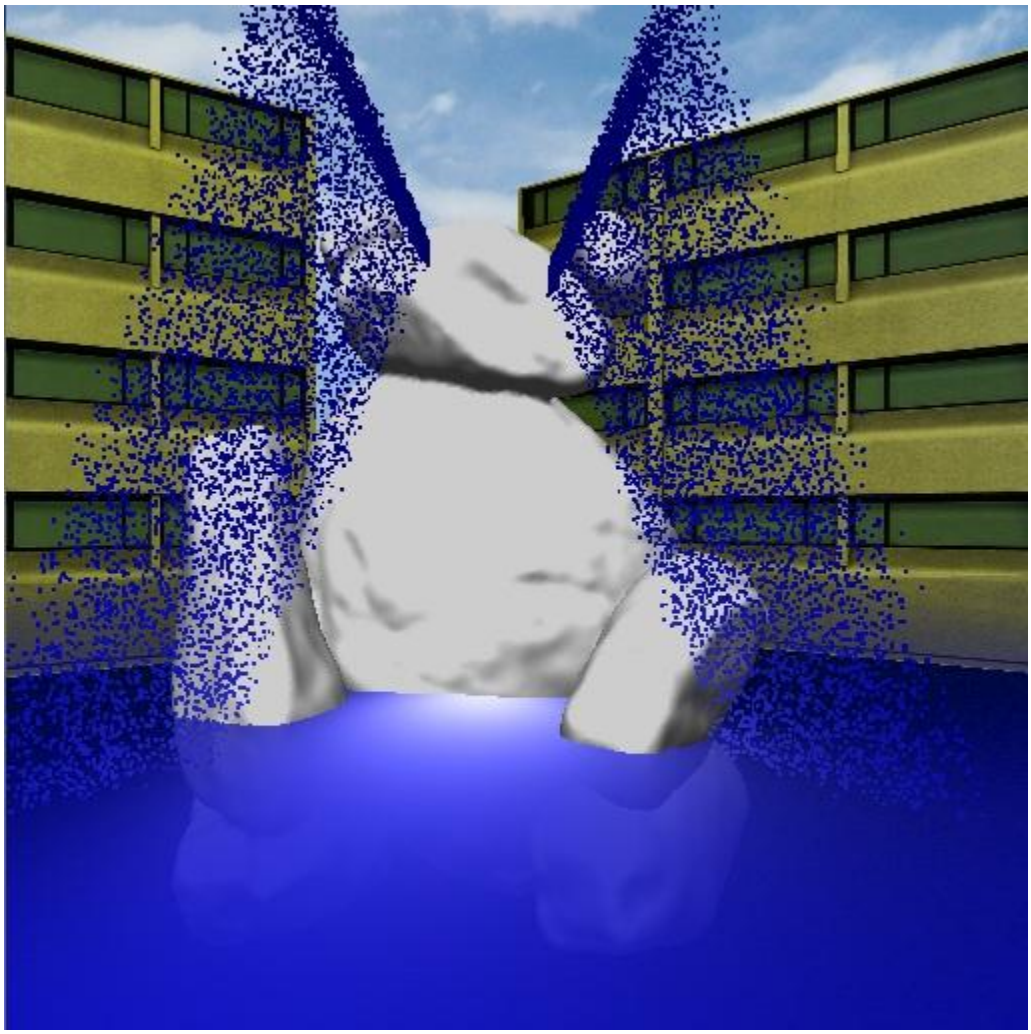
(picture of Bear with no shading)



(picture of Bear with point-per-pixel shading)

(picture of Bear with toon shading)

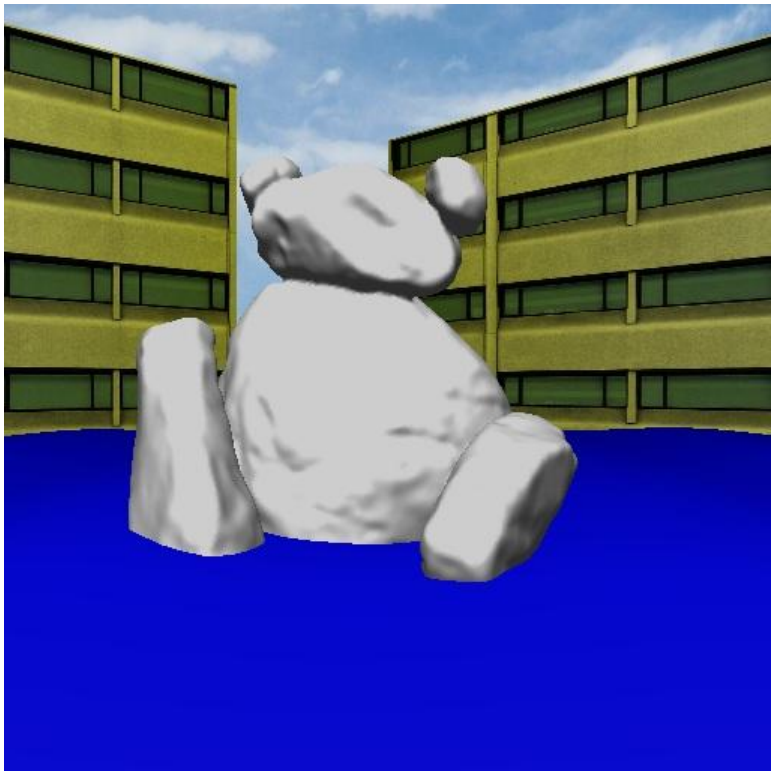
To give a reason for the sudden body of water being created, we came up with the idea to have the bear create the pool of water through the form of tears. For this reason, we introduce a particle system to simulate particles of water flowing into the body of water. To make it seem like particles of water, we use numerous random number functions to generate the directions in which the particles move. In addition, the life span and initial velocity of each particle varies as well to bring some randomness and spread out the particles. Certain thresholds are used to prevent the particles from flying all over the place or living forever. In addition, thresholds are used to make the flow of particles move in a certain direction and orientation.



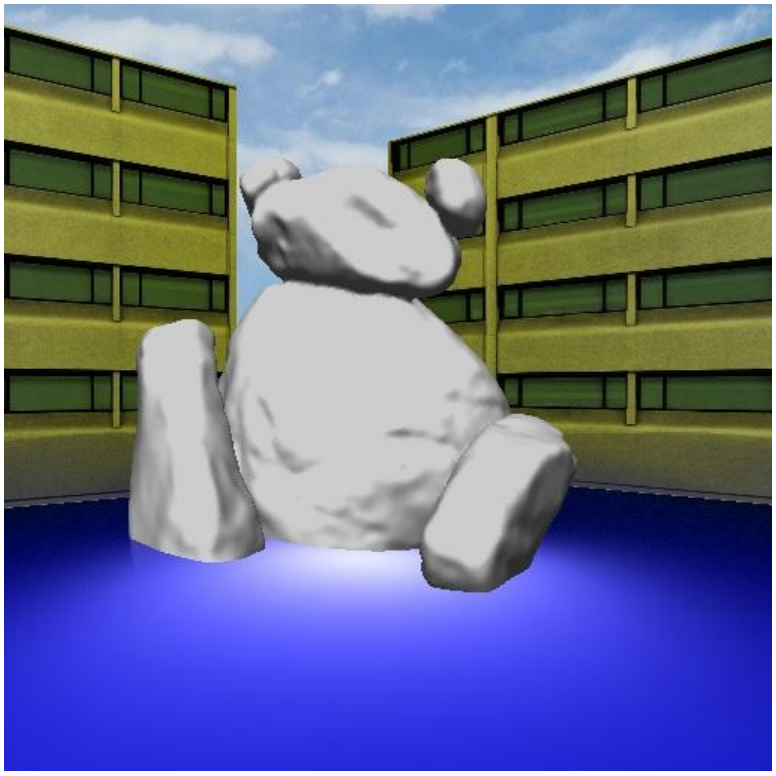
(picture of the Bear crying by particle system)

To complete our project, we worked on additional effects that portray the visual properties of water. We introduced a shader that contributes the Fresnel Effect to the surface of the pool of water. This effect is what causes water to seem both transparent and opaque. From a large view angle compared to the normal vector of the water surface, water tends to reflect a lot light making it bright and glare at the viewer. However from a small view angle, water tends to reflect little light, and the viewer sees through the water surface as if it is transparent. To simulate the transparency, we use

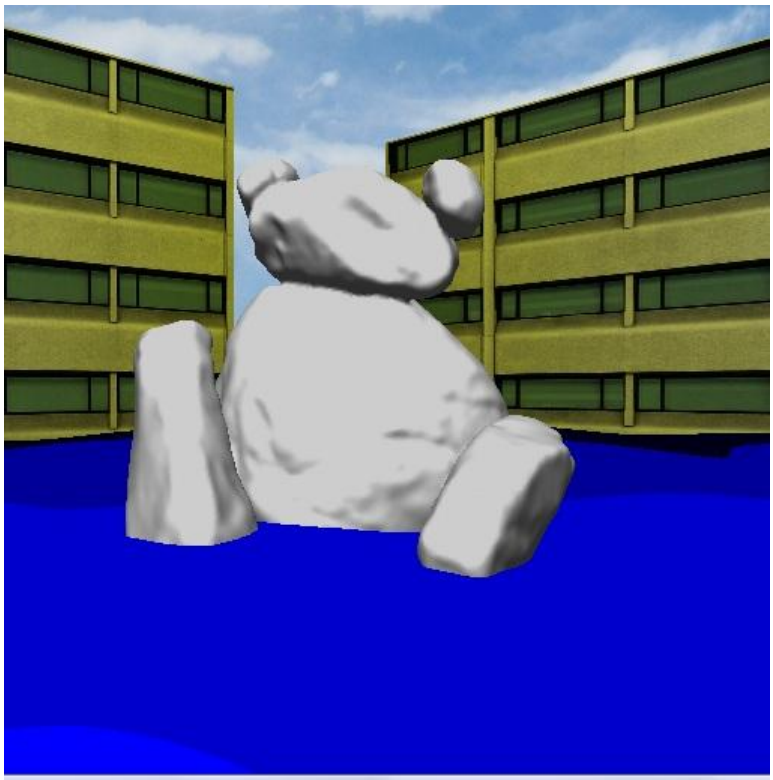
alpha-blending with the water to make it blend colors with the objects behind or within it and make it "seem" transparent.



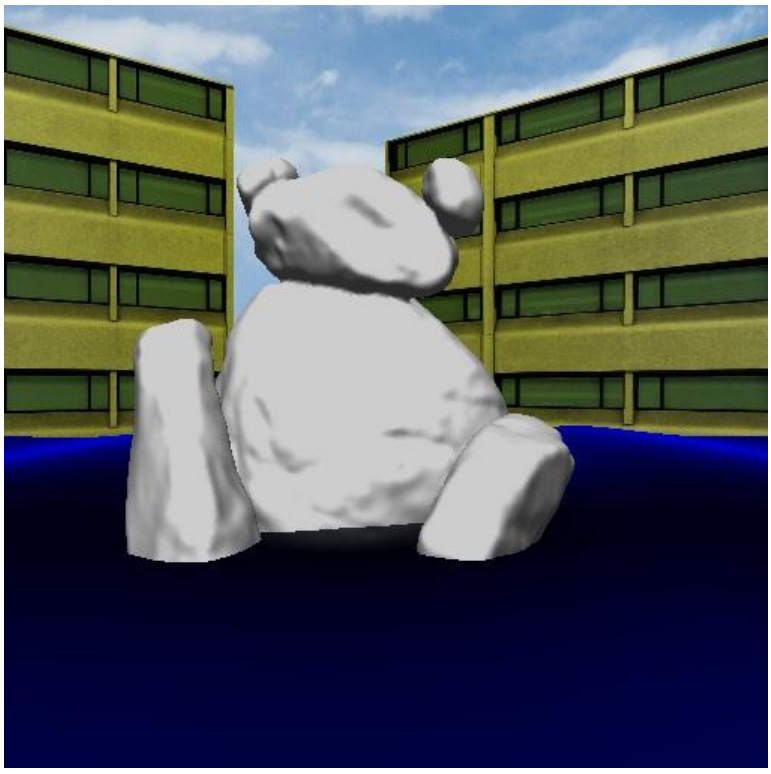
(picture of the body of the water filled half way to the bear with No Shading)



(picture of the body of the water filled half way to the bear with Point-Per-Pixel Shading)



(picture of the body of the water filled half way to the bear with Toon Shading)



(picture of the body of the water filled half way to the bear with Fresnel Effect Shader)

(picture of the body of the water filled quarter way to the bear with Fresnel Effect Shader)

[more specular version]

(picture of the body of the water filled half way to the bear with Fresnel Effect Shader)

[more specular version]
(picture of the body of the water filled quarter way to the bear with Fresnel Effect Shader)

There are some particular effects that we had not done in our project due to the intensity of the calculations. For the Bear's tears, we didn't handle proper alpha blending. If proper alpha blending was taken care of, it would project a visual effect similar to what is seen on waterfalls. Large masses of water particles with randomness of movement would contribute to a range of colors from darker blues to lighter blues to whites to demonstrate transparencies based on density of the water particles. However, based on the number of particles we used, we would have to keep track or calculate a particle's position compared to another water particle's position for proper alpha blending. This relative calculation would be on par to calculating collision detection between all the particles.

In addition, another intensive calculation would be calculating where the particles hit the body of water to produce a visual effect of ripples and turbulences in the water. This is literally doing collision detection for thousands of particles as they hit the water. In addition, the body of water is constantly moving, so it'd have to be calculated constantly. The region of contact of the particles with the body of water would gradually get closer to the Bear as the body of water rises up. Instead, we handle this by determining general locations to produce the ripples and turbulences based on the water level.

Program Usage:

Toggle Keys

- "C" - causes the bear to cry and stop crying
- "B" - toggles the bear between the various shaders (no shading, per-pixel shading, and toon shading)
- "W" - toggles the body of water between the various shaders (no shading, per-pixel shading, toon shading, and Fresnel Effect shading)