Brandon Caudell
EECS 672 – Computer Graphics
Project 2

Bowling Center Mock-up

Bowling has been a significant part of my life for many years now. As such, it was the obvious choice for a unique scene for me to model for this project. While I did not go all-out on my scene, I put a substantial amount of time and energy into modelling the scene as close to scale as I could. It features twelve lanes, along with all the appropriate pins, pin decks (the floor past the lanes the pins sit on), gutters, cappings, etc. I feel this model more than sufficiently meets the requirements of the project.

I started the modelling with the simplest structure: the lane surface itself. Using the Block class from the MandM example as a starting point, I created the lane by sandwiching 38 slim, black Blocks between the 29 boards of the lane (also Blocks). In hindsight, this model could have been much more efficiently done by creating one “block” with 77 rectangular faces on top, then assigning colors to each face using ebo's. Since this was the first part of the model, though, I took the route that I knew I could handle right off the bat.

The second model was probably the most complex, but it is the one that I was most excited to do. Modelling the pins was more of a challenge. I decided to treat them as a cylindrical object with a radius that varies as a function of height. At each half-inch “slice”, the pin is a circle with a particular radius, which I could find schematics for online. By approximating these circular slices, and then forming faces from the rings, I could create a fairly-accurate-looking pin model. I decided to leave the normal vectors as per-vertex attributes (rather than per-face) because this allowed the shadows to “blend” down the pin, rather than having each ring of faces be the same color with hard boundaries between. I treated the color attributes as per-face attributes, which enabled me to create the red rings around the neck of the pin without any linear interpolation of color. I then built up a larger model with ten of these pins arranged in the proper triangular arrangement, also to real-world scale.

The rest of the model was comparatively easier. The gutters use much of the same math as the pins, but the radius is fixed and only half of the cylinder is made. The cappings are simply Blocks of different widths. I then assembled a Pair model which builds two lanes (with gutters, pins, and pin decks), as well as the appropriate cappings. The “outer” cappings of the models are small, and the middle one is larger. In the real world, this is because the ball return runs under that middle capping.

Finally, I add six of these Pair models to the controller. There is a flag in the Pair model that turns on or off the left capping. This is because the right capping of one pair would overlap with the left capping of another. Thus, I disable the left capping in all-but the leftmost pair. This creates 12 lanes in a fairly-accurate model.

There is some debate to be had about how many types of models I use. Strictly speaking, I only add one class of ModelView to the controller, but that class contains many complex sub-models. Looking at the work of my classmates, I feel that my scene is more complex than many of the submissions of my peers. Because of this, I don't see any reason why I should be counted off for this requirement.

The only strange or difficult occurrence I encountered during this project was an obscure

“flipping” of the x-axis. For instance, the Pair model that I assigned the smallest x-value to appeared on the right of the scene (which I had expected to be the positive direction). I am still unsure what caused this in my models. Other than that, the project was relatively straight-forward. I was able to take enough cues from the MandM model to guide my usage of vao's and vbo's without feeling like I copy-pasted the code too much. Using the ebo's to create faces from specific vertices make the modelling very easy, and my background in mathematics (I'm double-majoring) made the calculations for the more complex models quite trivial.

The only items worth noting on my project are that I did not use the cryph utilities at all. I was able to use my geometry and linear algebra skills to compute coordinate and normal vector data directly. I admit that I did copy-paste the Block class from the MandM example, but this was such a primitive model that I didn't feel bad about using it to build larger models. Overall, I am very pleased with this scene. I gave the project the time and effort it deserved, and the scene I ended up with is fairly impressive.