Hayden Shelton November 1, 2015

**CS 4600 Assignment 4 Write Up**

Summary:

I chose to model a dog since I’m a dog lover. This assignment was somewhat challenging for me since it was my first time really diving into WebGL. The part I found to be the hardest was getting things to look right. It was difficult to get the ground plane correctly aligned with the dog’s feet and to perform transformations to the dogs body parts so that they were rotated and positioned like they might be on a real animal. That being said, I felt like I learned a lot and I look forward to shading this dog, I think the animation will turn out nicely.

Meeting Requirements:

I used example code from Ed Angel to draw the spheres, and then I figured out how to randomly color vertices by passing a buffer of random RGB colors to the vertex shader, which passes colors to the fragment shader. To model the various body parts, I applied transformations using my hierarchy diagram, so that each part is transformed relative to its parent node. I used non uniform scaling, translations, and rotations.

To make the dog walk, I set variables for X and Y position. These position variables are updated every 20 milliseconds when the scene renders, and change based on the direction slider. The X position is incremented based on cosine of the slider value and the Y position is incremented based on the sine of the slider value. The direction the dog is facing is changed by doing a rotation about the Y axis. The motion of the dog’s legs is approximated by taking the sine of the X position and translating by a multiple of that amount.

To figure implement the correct perspective, I looked at some of Ed Angel’s perspective examples and combined those with what I’d learned in class. I send the modelViewMatrix, the projectionMatrix, and the normalMatrix to the vertex shader as uniforms. The vertex shader multiplies the vertex position by these matrices to compute the new vertex position. To achieve the 80 degree frustrum, I use the perspective function because it was easier for me to understand. I provide sliders to adjust the eye position as well as the x position. Hitting ‘r’ will reset everything except for the speed and direction values.

Default Values:

For the default position of the eye, I chose (0, 0, 8). This gave me a good close up look at the dog. For the default position of at, I chose (1, 0, 0). Since the dog moves from left to right, it makes sense to look a little left of the origin. For the default value of the speed I chose 0.02. This means the dog’s x position is incremented by 0.02 every 20 milliseconds. This made for a nice normal walk. I set the near plane to be 7 and the far plane to be -12. This gives me a decent sized viewing volume.

Extra: Camera Tracking – You can toggle camera tracking by clicking on the blue button. It updates the x and z coordinates of the camera’s at vector.