CS405 – Project 2 Report

1-

For this task, I modified the else part of the set texture function by adding the following code:

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_WRAP\_S, gl.CLAMP\_TO\_EDGE);

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_WRAP\_T, gl.CLAMP\_TO\_EDGE);

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_MIN\_FILTER, gl.LINEAR);

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_MAG\_FILTER, gl.LINEAR);

gl.TEXTURE\_WRAP\_S and gl.TEXTURE\_WRAP\_T (X and Y respectively) allow me to clamp the respective coordinates outside of range [0,1] to be “clamped” (to lock in) in the range

The following gl.TEXTURE\_MIN\_FILTER and gl.TEXTURE\_MAG\_FILTER allow better “filtering”, however I am not sure if it is strictily necessary.

2-

For this task, I made the following changes, I am not sure if they are adequate but I believe they are appropriate:

For the fragment shader:

uniform vec3 u\_reverseLightDirection; (Variable introduced outside of main)

(within the if condition)

float ambient = 0.1;

vec3 normal = normalize(v\_normal);

float light = dot(normal, normalize(u\_reverseLightDirection));

gl\_FragColor = texture2D(tex, v\_texCoord);

This code allows the light values to be calculated within the fragment shader. These values are used as variables in other parts of the code. I borrowed, modified, and utilized the code from Week 11 recitation for this part.

For the constructor:

this.normalBuffer = gl.createBuffer();

this.enableLightingLoc = gl.getUniformLocation(this.prog, 'enableLighting');

this.ambientLightLoc = gl.getUniformLocation(this.prog, 'ambient');

this.normalLocation = gl.getAttribLocation(this.prog, 'normal');

this.reverseLightDirectionLocation = gl.getUniformLocation(this.prog, "u\_reverseLightDirection");

These values are values described in the fragment shader, but were implemented/imported by me later on. Normal buffer is initialized here (not in the fragment shader), which is used to hold the normal vector values.

For the setMesh function:

gl.bindBuffer(gl.ARRAY\_BUFFER, this.normalBuffer);

gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(normalCoords), gl.STATIC\_DRAW);

Within this function, I applied the same changes done to the other buffers, which is to bind the normal buffer and buffer the coordinate data onto the buffer.

For the draw function:

gl.bindBuffer(gl.ARRAY\_BUFFER, this.normalBuffer);

gl.enableVertexAttribArray(this.normalLocation);

gl.vertexAttribPointer(this.normalLocation, 3, gl.FLOAT, false, 0, 0);

gl.uniform3fv(this.reverseLightDirectionLocation, normalize([-lightX, -lightY, 5]));

This code is based on what we have done to the other buffers aswell, I have also utilized the reverseLightDirectionLocation value similar to how it was done on the Week 11 recitation, but I do not remember exactly why this operation was done.

For the setAmbientLight function:

This function takes the ambient value (defined by the input in the browser) and applies it to the ambientLightLoc value, with the idea being to manipulate the light value by the input.

For the enableLighting function:

Similar to what was done to the setAmbientLight function, we take the input from the browser and apply it to the enableLightingLoc value, with the idea bing to manipulate whether or not the light value will be enabled by the input.

Alper Büyükalaca

28978