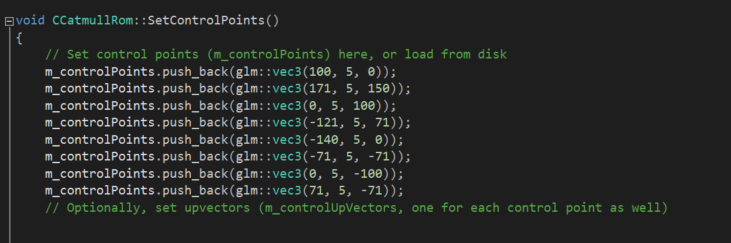
INM376:Computer Graphics Path Coursework Report

## Route

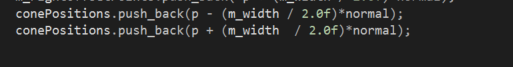
To create the path I followed the path starting guide and tutorials in class, which resulted in a path that is generated by interpolating a curve between a vector of points. With that curve I then build the track using a tnb coordinate system to add verticies along the edges of the track, add them to a vertex buffer object and render them as a triangle strip. The track is flat so all the normal are facing upwards.



I also added texture coordinates to the vertex buffer object so that the track can be textured.



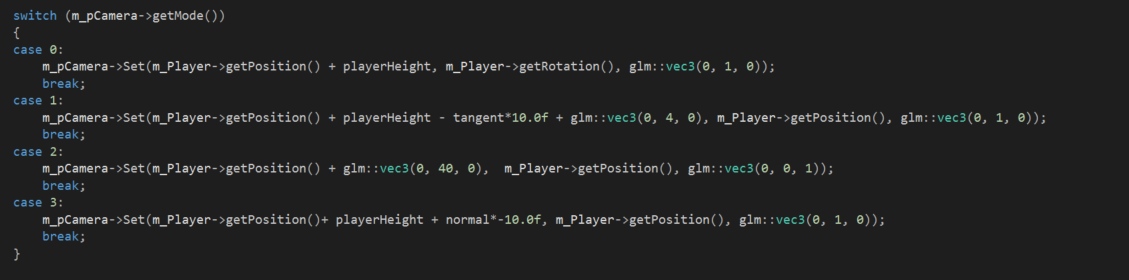
I Demarcated the edge of the track using a model of a traffic cone which I placed on the edge of the track.



## Camera

Camera view modes are changed by pressing number keys

Each of the camera modes relies on pre-calculated TNB frame vectors to know which way the player and the camera should be positioned and rotated.



In first person mode the camera is at the same position as the player, with the same rotation, and the player model is not rendered.

In third person the camera is behind the player, this is set by adding an offset vector to the player position and then making the camera look at the player.

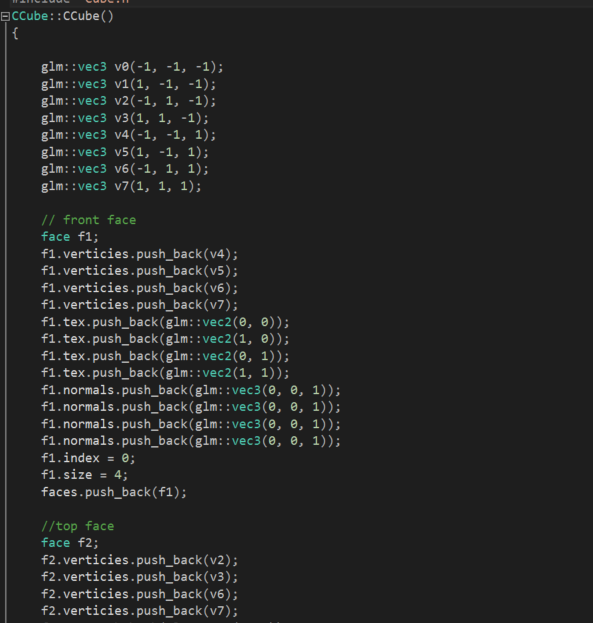
The top down view is achieved by making changing the up direction of the camera, adding a height offset to the player position and then making the camera look at the player.

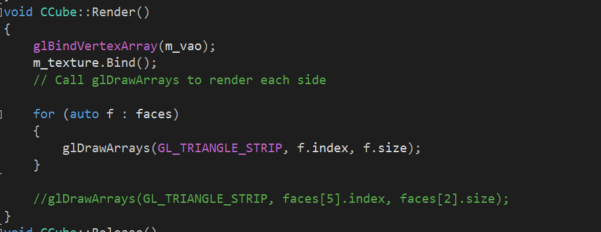
The side view is similar to the third person view but it uses the binormal rather than the tangent to add an offset of the player position. The camera is set to look at the player’s position.

## Basic objects

For the basic objects I created a class for the object and a face class that stored the data about each face. Each face contained a list of vertex positions, texture coordinates and normal directions. All of the data is added to the gpu as a vertex buffer object and then each face is rendered using a triangle strip.

The first primitive I tried to add was a cube, but I ran into a problem where the normals were not being set correctly and the some faces were being rendered backwards regardless of the direction of the normal of that face.





I then tried to make a 4 sided pyramid but I also ran into the same problem with that. I made a method in the face class that works out the direction of the normal based on the position of 3 of the verticies.

## Meshes

I added a cone, car, Stonehenge and well model to my scene.

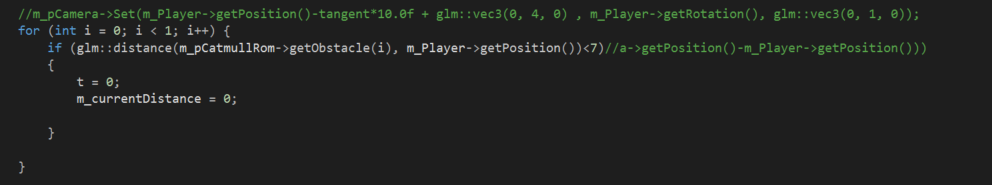
I translated, rotated and scaled the models and then rendered them in the scene.

The car mesh is used as the player avatar, the cones to mark the edge of the track and the other two are just to decorate the scene.



## Gameplay

The gameplay is endless runner style game, where the player must move the car left and right as it moves along the path to doge the barrels that are blocking the way. If the player collides with the barrels the player loses. I ran out of time before adding more barrels and so there is only one barrel on the path as of this time.



## Lighting

I added 2 spotlights to the scene, 1 white and 1 blue.

I also moved the lighting to the fragment shader to make the lighting effect smoother by applying the lighting per fragment rather than per vertex.

I made the light switch on and off by passing a float value to the gpu and multiplying the global light by it to lower its brightness. I found that turning it all the way off resulted in a poorer overall visual effect.

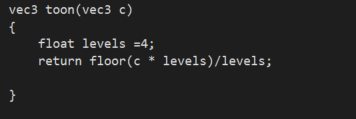
## Advanced Rendering

Implemented two advanced rendering techniques a toon shading effect and a fog effect.

The toon shader effect was achieved by quantising the lighting output based on the intensity of the light output.



To make the fog shader I made a fog method that changes the colour of the fragment’s output colour based its distance from the camera. I used a quadratic exponential function for the fog for more believable overall effect. The fog is applied to the skybox using a flat value so that the skybox is still partially visible as the fog would make it completely disappear.



## Controls

|  |  |
| --- | --- |
| Move forwards | W |
| Move backwards | S |
| Move left | A |
| Move right | D |
| Change camera mode | 1,2,3,4,5 |
| Turn off light | L |
| Turn on Light | K |

## Notes

When running in debug mode a problem with the well model causes the program to trigger a break point, press continue and the program will run correctly.

## References

<https://opengameart.org/content/lowpoly-car>

<https://opengameart.org/content/traffic-cone>

<https://opengameart.org/content/4-piece-stonehenge-opendungeons>

<https://opengameart.org/content/medieval-well>

<http://in2gpu.com/2014/07/22/create-fog-shader/>

<http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/toon-shader-version-ii/>