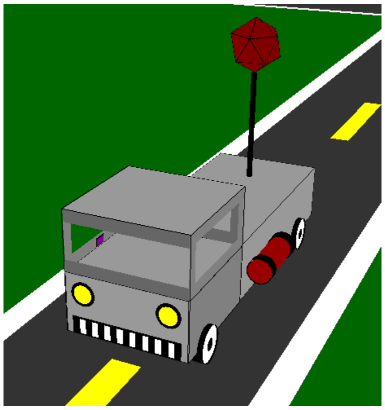

University of Lethbridge

3710 group project summary

Hummer Assault

Steven Deutekom
Lorenzo Conrad
Josip Smolcic
Dustin Ward

April 4, 2020

1 Introduction

This document contains a summary of information on our CPSC3710 group project. The project was to create an interactive 3d game in OpenGL to drive a hummer around a grid of city blocks. We were able to finish all the requirements for the project and had some fun doing it.

We will start by outlining the contributions of each group member. Following this we will discuss some of the design decisions and special structures that were used in the creation of the program. Then we will outline some of the things we learned, or couldn't figure out, with OpenGl. Finally, we will outline the assumptions we made in order to make the project more manageable.

2 Group Contributions

There were 3 group members. Below is a list with their names and their contributions. Everyone contributed to the coding, though each person had a section that they were most responsible for. We also all contributed to coming up with the graphics designs. We shared information and helped each other debug when necessary.

- **Steven Deutekom** was responsible for the overall class design and project organization. He organized code and added documentation to keep things consistent.
- **Lorenzo Conrad** was responsible for implementing some of the display methods for classes that were not the hummer.
- **Josip Smolcic** was responsible for implementing hummer code.
- **Dustin Ward** was responsible for getting the main GL routines and Game class implemented.

3 Design

The main design choice that we made was to make a GameObject class with a special interface for all objects that would be in the game world. All game objects inherited from this class and implemented a `display()` and `update()` method. This meant that in the main display function we could just call the display method for each of these objects to draw them, as well as teh update method to make any changes to the object's state every frame. We then tried to create a class for each object that we needed.

The main object was the hummer. This was the hardest to write all of the draw functions as it has a lot of parts. We put all of the different parts into their own functions to make it easier. All draw functions started at the origin and positioned themselves relative to the centre of the hummer. Then were translated and rotated to wherever the hummer was on the map. Because there is only a small area on the road where the hummer can turn we added some indicators that “light up” when the hummer can turn. There are 3 lights in the cab in each look direction. These have special colours to remind you which way you are facing. Also the antenna will get brighter when you can turn. When turning the hummer rotates around the centre of the cab.

The next most important part was our Block class. This creates a surface along with roads and intersections and random buildings. Each block is setup as described in the project description. By encapsulating them this way all we needed to do was create a grid of blocks, each starting at a given position so they overlap nicely. When a block is created it randomly decides to make 1-3 buildings. It also randomly decides which of the 3 building types to create. This ensures that each block is different, and that the game looks different each time it is run.

The roads are just two simple classes. One to be a straight strip of road with lines on the side and middle. The other to be an intersection with lines in the corners to match with the side lines of roads, but with no lines in the centre.

The buildings could easily have been 3 separate classes, but we just put them in one to keep it simple for now. The display function for each building is separate and we call the appropriate one from the `display()` method. We had some fun with the appearance of the buildings. We put 4 windows on each side of the rectangular building. We made the cylinder building multi-coloured and made the box on top open on the sides. Finally, we made the sphere have a fanned out wire cone that looks like a satellite dish rotating on top of it. We also added some wire loops around the centre for a futuristic feel, but also to better define the sphere visually.

Outside of the grid we drew a large flat brown area. Since the hummer is supposed to stop and be stuck if it is off the streets we imagine that the surface is mud. You go off the road too far and you get stuck. Then you have to move back by pressing the ’R’ key.

The camera positions were not easy to get setup. In the end we opted to keep a persistent LookAt object in the Game class that kept track of the position of the eye and where it was looking. This was used by all objects to display themselves properly. We set up tables that mapped a camera position and a direction to either an eye position or a position to look at. Then each frame we updated the LookAt objects position based on which camera we were using and the direction the hummer was travelling. This allows the camera to properly follow and react to the hummers movements and turns.

Finally, all the keyboard functions were set up in callbacks in the main file like in our other assignments. These checked some conditions that were tracked by the game like if the hummer could turn or if it was out of bounds, and if the game was paused. Each function key just changed the games camera position to be used with setting the LookAt.

4 OpenGL

We definitely learned things about OpenGl. We became more comfortable with how we would need to specify a sequence of primitives or predefined objects. Also, how to do this with the transformations for each object and resetting to the origin before each new object.

We also learned about some of the predefined structures like solid and wire cubes. It works quite nice when there are no shadows to use a wire and solid object so that there is some visual cue about the corners and edges of the object.

We made some silly choices, like building the cylinder tower out of line loops instead of an actual cylinder. This decision was made before we knew how to create cylinders, but we kept it because it made the colour variation easy.

Overall, we tried to use pre-made objects where it would save a lot of time typing out surfaces. However, when colour variations were desired or thing did not fit a nice pre-made shape we used the `GL_QUADS`. While sometimes time consuming a few extra objects or surfaces layered can add some nice visual effect. Though we are clearly still far from good at 3d design.

5 Assumptions

We did not have to make many assumptions, but there are a few things we did that are not perfect, even if they get the job done. For deciding when it is possible to turn we just mod the position of the cab of the truck and by the distance between the road centres. However, because it would be a very fine line we gave some grace. This means the hummer is not always in the perfect centre of the road. However, it does allow us to avoid collision detection.

We also, just used the position of the centre of the cab as the indicator that the hummer is out of bounds in the mud. Because the centre of the cab is not in the centre of the vehicle the position the hummer gets stuck is a little different depending on which side of the map you exit. It also matters whether you are travelling backward or forward. However, this is not a big deal in our opinion as the hummer does get stuck before travelling far from the road.

We also do not give any indication that the game is paused. Nothing can be done but un-pause the game when it is paused and all moving objects stop moving. So we feel it is good enough for now. It probably wouldn't be too hard to add these things, but for a game with nothing, but driving it is probably enough.

We also could not figure out how to write anything on the licence plate of the hummer in 3d. So it was left blank. All the ways we found online that might do this were from third party libraries. So we just left it blank.