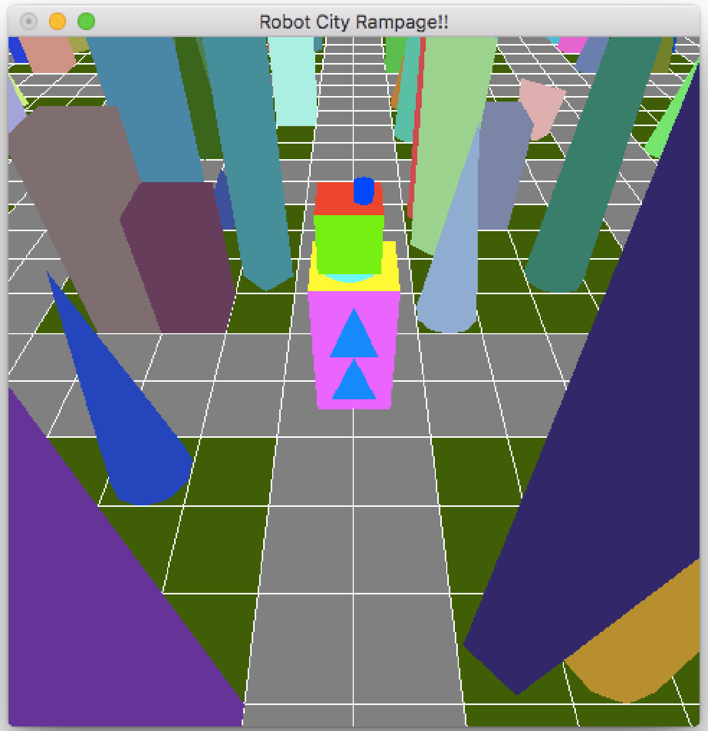

CPSC 3710 Project - Robot City Rampage

10.04.2017

Team Members

Mitchell Wever

Austin Kothig

Lukas Grasse

Design

The following are initial design ideas for how this program will be structured.

We started out with establishing these classes and functions:

Grid Function

function to build the initial 20x20 “block” grid where we established that grey squares will be the squares that the robot will be able to walk on. Then the squares that the blocks will occupy will be green. The streets will be three squares wide. The blocks will by 4 by 4 squares. The dimension of each square will be 25 by 25.

RealEstate Function

Function that will establish whether a square in a block will be occupied by a building. We will establish a 3d array with 2 levels in the third dimension where if in the first and second dimension, if a square is set to -1, then a building cannot be built there. If it is any other number, that enum will be related to a color and the number occupying the third level at that location in the 2d array will represent the type of building to be built.

Building Factory

Will build and return buildings that will occupy the squares where the real estate will have information for how that building will be built or whether a building will be built there at all.

Robot

Will build the robot will specified parameters where the only variability on the robot is it’s “eye’s position”, meaning the position of where the eyes are on the robot’s head so we don’t actually have to move the entire robots head. Then the robot can be built facing any direction and can be rotated accordingly or moved along the streets. We will be sure that the robot can only occupy one square at a time.

Lookat Function

The lookat will be established and change within the main file and will change according the key and mouse entries.

Click Events

We will also have the mouse selection ability to destroy buildings, but not destroy the robot itself. The robot does not have to be looking at the building to destroy the building, it simply has to be within the player's viewing area in order to destroy a building.

Assumptions

1. Assumed that you cannot see the robot through buildings.
2. Cannot destroy a building if it is not visible.
3. Robot does not have legs because it is from another planet and has the ability to move without legs.
4. If the robot exits the landscape, it can be reset using the r key. We assumed we did not have to implement the robot to stop before exiting the landscape.

Special Data Structures

Array Tables - for the body position, as well as the change in the view.

Used the Factory Design Pattern - For random building creation.

Vectors of pointers to buildings.

Distribution of Work

Austin

Worked on the button configurations (Special Key and Keyboard input). Worked out the numbers that should be in the lookup tables for the view change, as well as the logic for when a robot is 'legally' allowed to make a rotation, and the direction that they should appropriately move to based upon their direction. Worked out how the general landscape is made up. Worked out the generation of city blocks being independent from the roads.

Lukas

I implemented the generation of buildings, including their shapes, sizes, positions, and colours. I also worked on handling the click event of clicking on a building and figuring out which building was selected. I also implemented the destruction of buildings, including projectiles and the explosion of the buildings.

Mitchell

Maintained the major portion of the documentation parts of the project. As well as programmed the structure of the robot and how the robot is being constructed. I also helped in the initial design phase in terms of planning out how our whole program would work and look.

New Things Learned about OpenGl While Working on the Project

- Using selection and picking in Opengl to implement the destruction of the buildings.
- We learned about how to generate a grid to help orientate ourselves around the world as we slowly built up the project.
- We learned about how to make cylinders and how to cull them properly. In the same respect, we also learned that you can use the cylinder function to create cones by specifying the top base as a 0 radius.
- We also learned about the importance of specifying the order of when the objects in opengl are created and how that affects culling.
- We also learned a lot about how to properly use the PushMatrix and PopMatrix functions to their full potential in order to manipulate the environment and objects within it.
- Learned how to implement explosions into our project for when the building disappears.
- In addition to the explosions, we learned how to implement a projectile that will move towards a specific location when an event like a click takes place.