Programming Project Report

Pablo Guerra
010849976

Problem Statement:

For our next project, we were tasked with the job of designing and implementing a video game called “angry bricks” modeled after the game “angry birds”. This project was split into two parts to simplify the development process. The first part of the project was to develop the basic foundation of the game which includes the visual design, as well as obtaining user inputs. The second part involves simulating and display the motion physics.

Design:

In order to make the game appealing, a 3D cube was used for the interactive object as supposed to a simple rectangle. The cube was rotated slightly to allow visibility of three sides of the cube, making the visuals look more interesting and complex. However, this rotation did introduce an issue with the cube being redrawn slightly off from where the user clicks the screen. This issue was solved first translating the cube and then rotating it before being displayed. Another important part of drawing the cube is deciding which sides of the cube to draw in which order, since drawing sides in any random order would make it not look like a cube. This was initially solved by simply drawing the back side of the cube first and the front side of the cube last but was then updated by simply using the GL Depth Buffer.

Implementation:

The development of the project began by using the provided “cube2.cpp” sample code, which was modified to show a single cube at a fixed location on the screen using the display callback. After this was done, the next step was implementing the mouse callback which redraws the cube at the location the user clicks. This was the step that took the longest to complete as the rotation of the cube affected the position it was drawn. The final step was to add the ability to drag the cube in the window using the motion callback. This step was very straightforward as all the necessary foundation was done in the previous parts. Next, the initial points of when the cube is clicked and the final points of when the button was released were used to obtain a vector to determine the direction and speed of moving the cube. Certain values were chosen by trial and error for both the speed and rotation of the cube as well as the gravity effect.

Testing:

The program was tested after completing each individual step to ensure that any bugs introduced in that step were solved immediately. This allowed for easier debugging and changes to the newly implemented functions. Since there isn't much inputs for the program, it was easy to test all the cases and see what needed to be fixed. Each step of the project was not started until all the bugs were fixed for the previous steps. Below are images of interactions with the program.

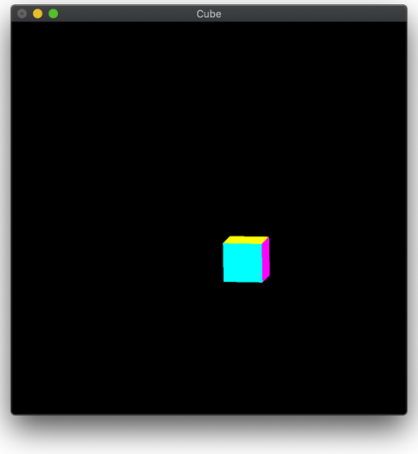

Image shows cube initial location

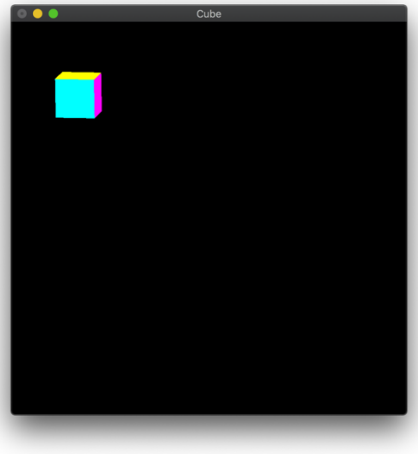

Image shows cube at clicked location

Conclusions:

Overall, the program works as intended and the user can properly throw the cube around in the window. All bugs found in the first part of the project was fixed and the remainder of the project was completed. The cube seems to have a natural motion and correct trajectory. There are, however, some occasional issues with the motion of the cube being inaccurate when it is launched on the edge of the screen due to the bouncing effect of the walls. This was not fixed and still remains an issue.