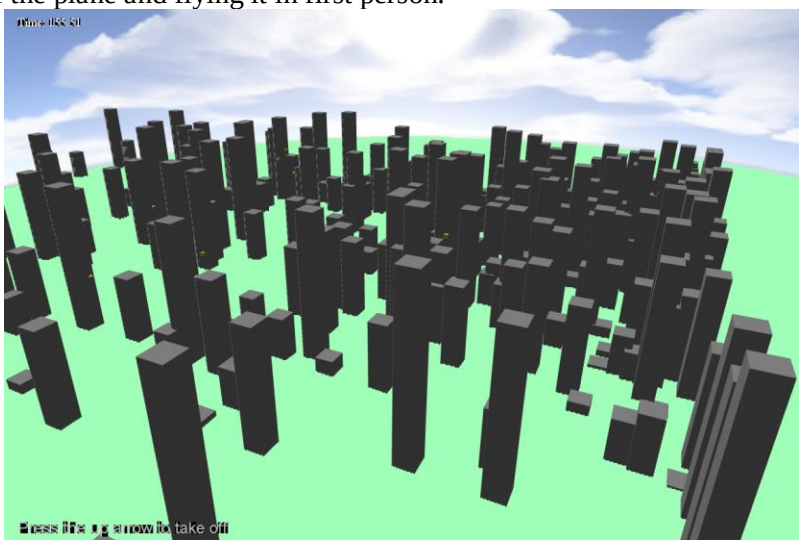
Keith Livingston
Donald Spickler
5/15/18
COSC 482

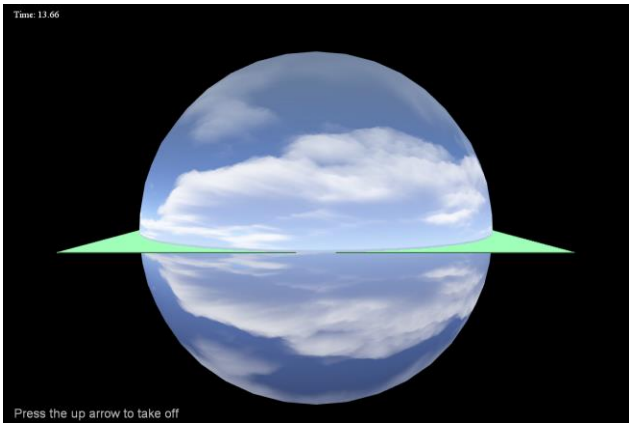
Flight Simulator 2018

This program is a flight simulation through a virtual city. When the program starts the user will be sitting stationary in the hanger. A randomly generated city will spawn in front of him with buildings and objects to interact with. When the user is ready they can take off from the hanger and fly into the scene. The user will control the direction of the plane with a set velocity after the user takes off. They will have the feel of being in the cockpit of the plane and flying it in first person.



When the program is first run the user will be sitting stationary in the Hanger. The plane is not actually stationary, but is just being stopped by an object in-font of the camera. The Hanger and Camera object are sitting on a plain that covers the entire scene. The Buildings are randomly generated at the start of the program. At any time during the program the user can press the 'R' key and reset the camera position back to the hanger and generate a new city. When the user is ready they can read the text at the bottom of the screen prompting them to, "Press the up arrow to start". The camera or airplane will now be completely under the users control. The airplane has a set velocity and the arrow keys control the movements of the plane. If the user wants to add Yaw to the camera or "barrel roll" the plane they can Yaw left with the 'A' key, or Yaw right with the 'D' key. Just like a real airplane the camera can tilt up and down faster than it can turn right and left. If the user wants to stop the airplane they can hold the 'alt' key and the velocity is stopped, the airplane will also slowly fall to the ground because there is slight gravity on the airplane. Since all the objects are created by the Bullet Engine they can all interact with each other. When the user starts there is also 10 gold spheres and 10 gold boxes that spawn randomly around the scene. When the user runs into any of the spheres or boxes they will be shot the other direction. When the user runs into a building or the ground the camera object will slide off it or just run straight into it. The whole scene is wrapped in a cube map that simulates a sky (Example in the image below). The whole city and world is wrapped in this cube map being mapped onto a sphere object.

This program was written in CodeBlocks in C++. Other libraries were needed to make this work. This program uses libraries such as, glu, sfml, Bullet Physics, and ming. All the objects in the World of this program or the scene are created by the Bullet Physics engine. This means that all the objects are "Bullet Objects" allowing the engine to interact with the objects in the scene. All the collisions are dealt with by Bullet and all the objects in the scene interact with each other in some way. The camera or airplane is a Bullet object that can run into the other objects in the scene because it is the only moving object. When the camera runs into a building it has a lower mass and will not move the object. If the airplane runs into a gold sphere or gold box it will knock it out of the way. The whole scene is wrapped in a cube map that encompasses the whole city and the bounds of the ground. The cube map is not a Bullet



object, so the camera and gold objects can fly straight through it. There is textures loaded into the program for the object but that feature is not working properly. These textures can be found in the textures folder for the project. Across the top are the name of the program, frames per second, number of objects in the World, and a brief instruction manual on how to fly the plane. The time is tracked in the top left corner of the screen and the instructions for how to start the plane or take off are at the bottom of the screen. All of the keyboard functions are dealt with in the UI files. The velocity of the plane is also set here. When the city is randomly generated it is designed to have a more congested "Downtown" area and as the city gets wider the buildings will become scarcer. This gives the real feel of a city with a downtown and suburb area. Since this is a simulator the flight should feel as real as possible. When the user presses the up or down arrow keys they will turn or tilt faster than the left and right arrow keys, much like a real plane. So, Yawing the airplane to the left and tilting up will make the user turn left faster than the left arrow key much like a real airplane. There is also a screenshot feature in the program allowing the user to capture the current screen with the 'F10' key. This is what I used to capture the pictures in this program. Each time you use this feature it saves the picture in the file "screenshot1" in the project folder. Only one screenshot may be saved here at a time so taking a second screenshot will replace the one that was previously there.