CPSC 3710 - Course Project

An Interactive Game using OpenGL

50 marks

Due: to be discussed in class

Project goal

In this project, you are required to design a simple interactive game using OpenGL. Imagine that you are guiding a robot through streets (with buildings) and explore them. During this process, you may also select some of the buildings and destroy them. In addition to the OpenGL functionalities you have learned in class, you are further required to read some more samples posted on the course website and apply the techniques there in this project.

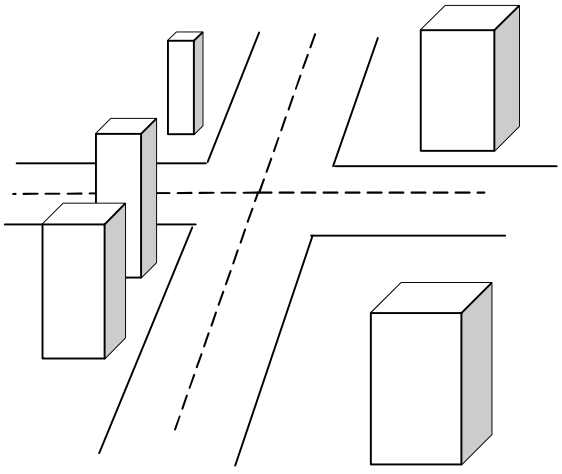
Also please explore the OpenGL links posted on the course website. It would be unrealistic for us to learn every single function in OpenGL in our class. Therefore this project will give you an opportunity to learn more about OpenGL.

You can form a group of at most four students and every group member will be awarded the same mark. It is group members' responsibilities to distribute and share the workload.

Project details

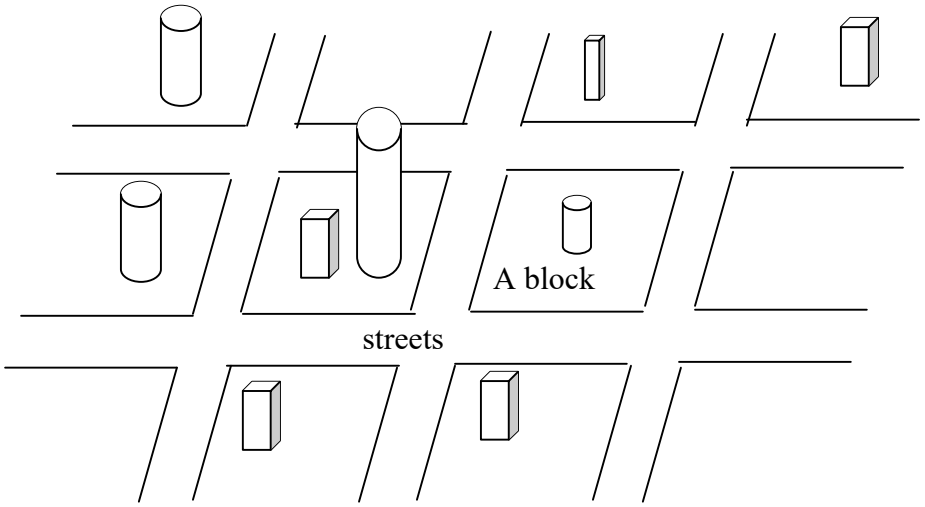
(1) Streets: Look at the following picture for an illustration of streets. You may assume that the robot is walking on the xz-plane. All the streets are parallel with the x-axis or z-axis. Note that the following picture only shows a partial map. As your robot moves around, parts the map will appear/disappear.

You assume that the robot can only move on the middle lines (as indicated by the dotted lines) of the streets. It can move forward, turn left and turn right. However, turning left and right can only happen at the intersection of the dotted lines (this assumption makes the programming much easier). If you try to guide the robot to turn not at an intersection, the command is ignored.



All the streets should have the *same* width and you decide it *appropriately*.

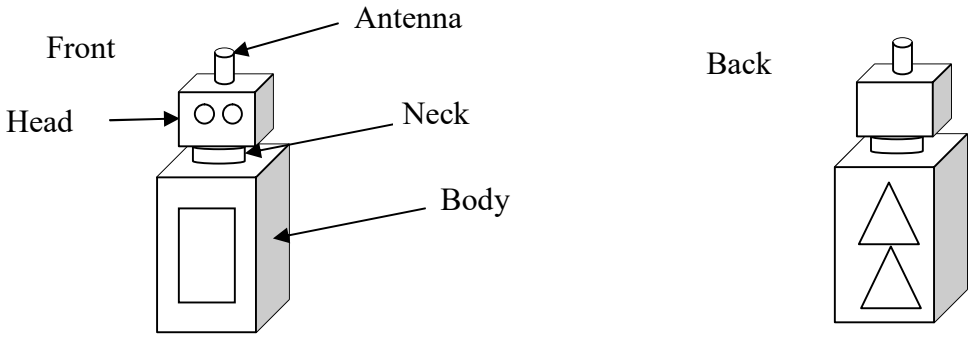
(2) Buildings: Imagine that your robot is exploring the downtown of the Manhattan area. The intersections of the streets form square blocks, as shown in the above picture. Again you decide a block's size *appropriately* and all the blocks should have the same size. There is a total of 20 x 20 blocks.



Within each block, randomly generate some buildings. Of course, a vivid building needs more efforts. But in this project, some prisms, cylinders, spheres etc. should be fine. You can add windows to buildings but that is not required.

There are three types of buildings. (1) Non-destroyable buildings; (2) Strong buildings; and (3) Easily-destroyable buildings. The types of the buildings are generated randomly.

(3) The robot: It has the following figure. Initially, the robot is at (0, 0, 0) and facing the negative z-axis.



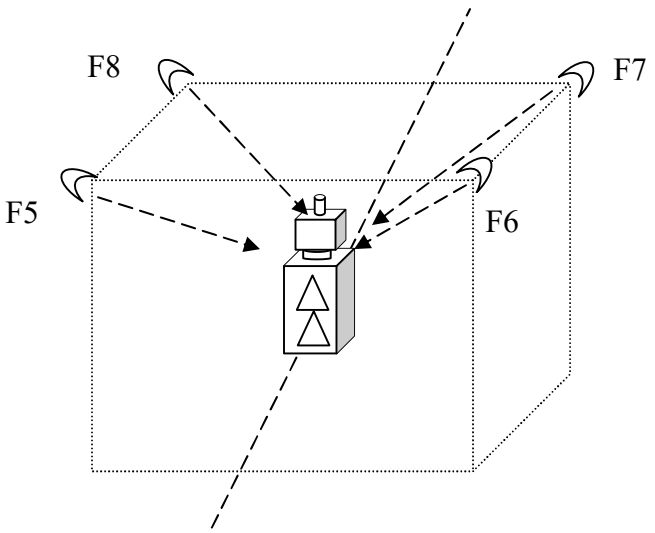
Your robot has two eyes and an antenna on its head. The antenna always rotates clockwise, with each step of 30^{o}. The robot's head can turn right, turn left, or face forward. You decide the proper size of

each component in the robot and can add some extra components to make it look like, say, the one in *Star Wars*. But the components in the above figure should be there.

Your LookAt view always follows the robot, i.e., the distance between your eye and the centre of the robot is constant.

(4) Actions: The following is a list of actions that you need to do.

(a) F1: turn the head of the robot to face forward. That is the default LookAt view.
(b) F2: turn the head of the robot to right. When F2 is unpressed, the head of the robot turns to forward.
(c) F3: turn the head of the robot to left. When F3 is unpressed, the head of the robot turns to forward.
(d) z: push the robot forward
(e) q: at an intersection of the streets, turn the robot to left. If the robot is not at any intersection, nothing happens.
(f) a: at an intersection of the streets, turn the robot to right. If the robot is not at any intersection, nothing happens.
(g) F5, F6, F7 and F8: install the LookAt view at the respective locations as shown in the following picture.
(h) F9, F10, F11, and F12: install the LookAt view at the directions similar to those in (g). But they are distant. This way you can see more of the scene around the robot, i.e. some of the buildings and streets around it. You decide the proper distance.
(i) F4: returns the LookAt view to its default setting.



(j) Left mouse button: move the mouse pointer over a building and click the left mouse button. The building could disappear. See below for more details. Of course, if the mouse pointer is not on any building, nothing happens. Please see the tutorial posted on the course website on how to select an object in OpenGL.
(k) p: pause the game. Pressing p again continues the game.
(l) r: returns the robot to the origin if it is on the boundary of the exploration area you created.
(m) If the robot is already at the boundary, any pushing is ignored.

Project assumptions

It would be unrealistic to design and implement a complete interactive game in a one-semester course. The following assumptions are made to make this project simpler.

(1) You decide the sizes and colors of different objects, including buildings, streets, blocks, etc. But they should be realistic to be visible.
(2) You can decide the number of buildings in one block, say 5, and fix it for every block. But every block should have some buildings.
(3) If a building is non-destroyable, no matter how many times it is shot, it will stay and will not disappear. If it is strong, it needs to get *three* shots before it disappears. If it is easily-destroyable, *one* left-mouse click on it will make it disappear.

You can also make any other *reasonable* assumptions, and if so, should document them.

Project submission

(1) Design and implement a solution to the above project using C/C++. Make sure that your project is compilable and executable in the computing environment of our labs.
(2) For your project, write a small tutorial (submitted as a hardcopy) as how to compile and use it. Your marker will check it based on your tutorial.
(3) Write a summary (hardcopy and at most three pages) of your design and implementation, including
    (a) The distribution of work among group members, with their names and duties.
    (b) Any special data structures used in your implementation.
    (c) Any OpenGL you have learned during this project (not the ones we discussed in our lectures or posted tutorials).
    (d) Any assumptions of your own you have made in your project.
    (e) Any other documentation that you think may help understand your efforts.
(4) Submit your project (all the source code files, make file, etc.) in a USB stick, such that the marker can check it. The date and time will be TBA. The USB sticker will be returned back to you after the project is marked.
(5) Do a demo, whose date and time will be TBA.

Project hints

(1) Implement the shooting functionality at the last step.
(2) Decide the centre of the robot and then add in the other components of the robot.

There might be some necessary adjustments to the project, if needed, in the future.