Assignment 1

Advanced Graphics, Augmented Reality, and Virtual Reality

August 2020

1 Learning Objective

In this assignment, we will build a 2D game. In doing so we aim to get comfortable with making objects of greater complexity with OpenGL, in addition to working with many interacting objects, and understanding how applications are built with OpenGL.

2 Problem Statement

In this task, you will be required to create a 2D maze completion game - inspired by the PacMan maze game, with a modern twist. To aid you in getting started, you can choose to build atop the simple template given alongside the previous assignment, or the one linked here.

The objective of this game is to exit a dark maze you have been dropped into. You have a lamp, which allows you to see a small part of the world around you, but bear in mind that the light cannot penetrate the walls of the maze. You must hurry, for something sinister is on it's way to get you.

Figure 1: A starting point to give you an idea of the game

1. World : A procedurally generated maze (different maze layout on every game startup) with no ambient light. Maze needs to have a single exit, and if player reaches this point, the player has exited the maze. The world can have 'details', but only in the presence of a light source can these be viewed. Implement a smoothly transitioning zooming camera, that can get closer to the player to zoom in and further away to show a larger portion of the map.
2. Player : A single player with some fixed health that can navigate the generated maze with the keyboard. Carries a lamp which when charged illuminates a small area around him. Over time, the battery of the lamp dims, and the visible area around the player slowly reduces and diminishes. The player can use the maze to evade the enemy.
3. Enemy : An enemy, somewhere on the map, which chases the player in the maze. If possible, it picks the shortest route to the player. If it catches the player, the game is lost.

4. Obstacles: Small obstacles in the maze that the player must avoid, and reduce the players health if it comes in contact with these obstacles. You can be creative in creating these obstacles.
5. Powerups : Batteries/Oil scattered around the maze to recharge the lamp. Any reasonable power-up to contribute to the score, or to aid the player in some way (eg. a temporary speed boost, health etc.), can be added as well. You are free to use or own creativity and imagination. These powerups will only be visible when within the radius of the light of the lamp.
6. Scoreboard : Keeps track of the score, player health and battery charge always visible on the screen. Feel free to decide the scoring metric yourself.

Other objects, and complexities can be added as well. Be sure to implement proper and accurate collision detection. Bonus points will be given by the TA if something impressive is added. This can be carried forward to other Assignments only. Don on your Thinking Hat :)

3 Hints

1. Following OOP principles might be very helpful as your game scales. When you create objects in the world, make them as objects in the program (as classes) with all the parameters and methods needed for drawing, motion etc. built into the object along with a method to draw themselves.
2. While very realistic lighting will not be a requirement in this assignment, and you can get by with using simple 2D masking tricks, we encourage you to explore different ways of lighting your scene. OpenGL libraries take care of a lot of the internal concepts for you, and you can apply some of the lighting effects that you have learnt about in the course. Here are a couple of resources to get you started.
   Lighting Theory + Example.
   OpenGL: Basic Lighting.

4 Marking Scheme

- World : 20 points
- Player : 15 points
- Enemy + Obstacles : 15 points
- Batteries/Power-ups : 10 points
- Lighting : 10 points
- Scoreboard, and Battery indicator : 10 points
- Gameplay : 20 points

5 Submission Guidelines

You are free to use any low level implementation of OpenGL. The assignment can be written in C++ or Python. You should provide a simple single-page quick start guide outlining the game controls, and any additional features. Also provide build instructions for your game. Please submit all your files in a roll numbered zip on Moodle on or before 11.59 PM August 30, 2020. Keep in mind the late day policy, and use them wisely.

Best of luck!