Software Requirements Specification

for

Beyond The Light

Version 1.0 approved

Prepared by Karen Salinas

CMPS3350

7/2/17

Table of Contents

1. Introduction

1.1 Purpose

The purpose of Beyond The Light is to modify a given framework developed in the course and elaborate the outline of the specifications of the game development. Beyond The Light is heavily influenced by the "Walk" framework and inspired by the games "Luigi's Mansion 2D: Eternal Night" and "Slender Man", in which is intended for teens ages 11+.

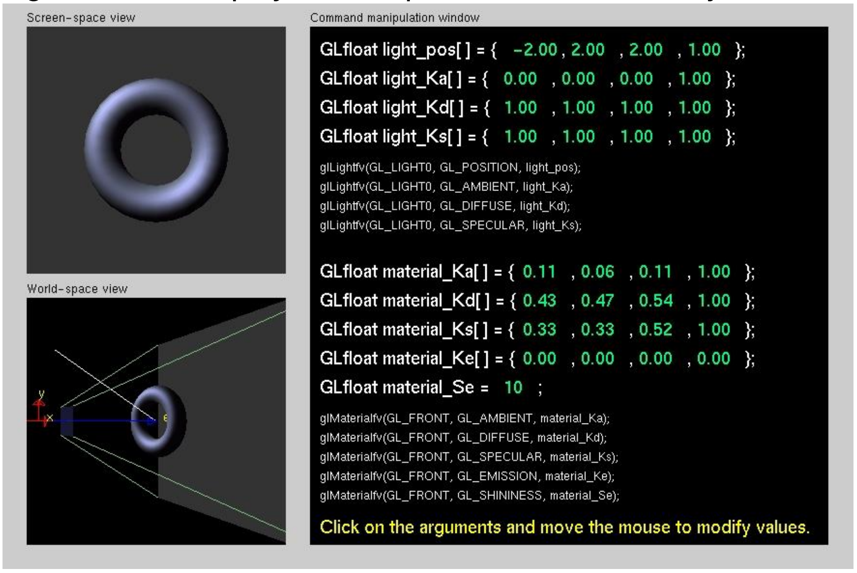
1.2 Document Conventions

The document follows with C++ programming language, and OpenGL functions in which some terminology may be unfamiliar to some readers. Most of the terminology used are prototype functions using void, and OpenGL implementation such as glEnable, glClearColor, and so forth. The list of definitions used in OpenGL can be found on websites that list out all the libraries and application programming interfaces. A reference to consider for looking up the definitions for OpenGL is:

https://www.khronos.org/registry/OpenGL/api/GLES/1.0/gl.h

1.3 Targeted Audience

The document is written for students, software developers, and programmers who are learning OpenGL and game development. Beyond The Light offers OpenGL components paired with C++, in which students who have taken previous programming course can develop from the familiar programming language to create a modified version of the framework. The figure below displays how OpenGL renders an object in code.



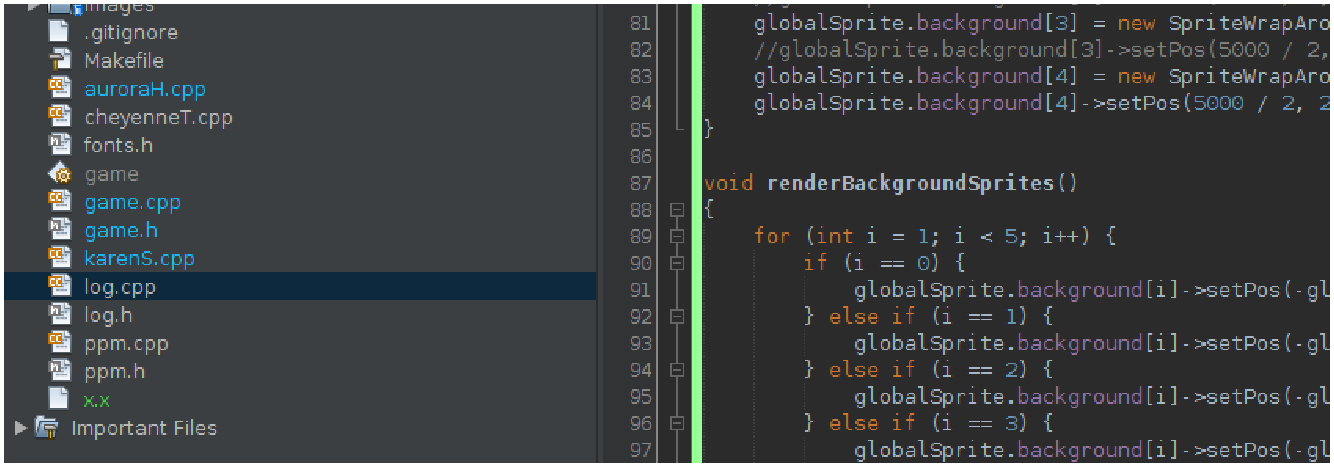
```
GLfloat light_pos[ ] = {  -2.00 , 2.00  , 2.00  , 1.00  };
GLfloat light_Ka[ ] = {  0.00  , 0.00  , 0.00  , 1.00  };
GLfloat light_Kd[ ] = {  1.00  , 1.00  , 1.00  , 1.00  };
GLfloat light_Ks[ ] = {  1.00  , 1.00  , 1.00  , 1.00  };
glLightfv(GL_LIGHT0, GL_POSITION, light_pos);
glLightfv(GL_LIGHT0, GL_AMBIENT, light_Ka);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_Kd);
glLightfv(GL_LIGHT0, GL_SPECULAR, light_Ks);

GLfloat material_Ka[ ] = { 0.11  , 0.06  , 0.11  , 1.00  };
GLfloat material_Kd[ ] = { 0.43  , 0.47  , 0.54  , 1.00  };
GLfloat material_Ks[ ] = { 0.33  , 0.33  , 0.52  , 1.00  };
GLfloat material_Ke[ ] = { 0.00  , 0.00  , 0.00  , 0.00  };
GLfloat material_Se =  10  ;
glMaterialfv(GL_FRONT, GL_AMBIENT, material_Ka);
glMaterialfv(GL_FRONT, GL_DIFFUSE, material_Kd);
glMaterialfv(GL_FRONT, GL_SPECULAR, material_Ks);
glMaterialfv(GL_FRONT, GL_EMISSION, material_Ke);
glMaterialfv(GL_FRONT, GL_SHININESS, material_Se);
Click on the arguments and move the mouse to modify values.
```

Figure 1

1.4 Product Scope

The product scope specification establishes the functional, performance, and development requirements for Beyond The Light. The purpose of the project is to create different source files from individual members within the group to integrate into one single main file to execute as one compiled game. In the following, the demonstration the project's files within the program asset are neatly organized.



```
images
.gitignore
Makefile
auroraH.cpp
cheyenneT.cpp
fonts.h
game
game.cpp
game.h
karenS.cpp
log.cpp
log.h
ppm.cpp
ppm.h
x.x
Important Files
```

```
    globalSprite.background[3] = new SpriteWrapAro
    //globalSprite.background[3]->setPos(5000 / 2,
    globalSprite.background[4] = new SpriteWrapAro
    globalSprite.background[4]->setPos(5000 / 2, 2
}

void renderBackgroundSprites()
{
    for (int i = 1; i < 5; i++) {
        if (i == 0) {
            globalSprite.background[i]->setPos(-gl
        } else if (i == 1) {
            globalSprite.background[i]->setPos(-gl
        } else if (i == 2) {
            globalSprite.background[i]->setPos(-gl
        } else if (i == 3) {
            globalSprite.background[i]->setPos(-gl
```

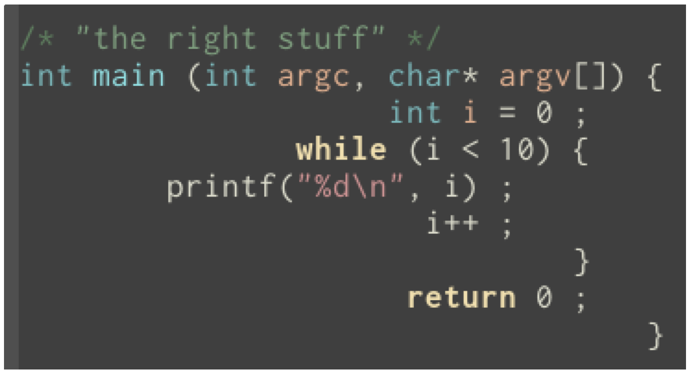
Figure 2

1.5 References

This document builds on the following references:

- An introduction to OpenGl cross-platform application programming interface[1]
- Modification of C++ programming language [2]
- K&R indenting style applied to all documents[3]
- Team communication over game development[4]

K&R is heavily used in the game development of Beyond The Light, as displayed below.



```
/* "the right stuff" */
int main (int argc, char* argv[]) {
                    int i = 0 ;
               while (i < 10) {
        printf("%d\n", i) ;
                      i++ ;
                             }
                     return 0 ;
                                 }
```

Figure 3

2. Overall Description

2.1 Product Perspective

Beyond The Light, the user walks through a series of interconnected frames by conducting a series of key strokes to help the character jump, walk left and right, with her weapon (sword/flashlight) to kill/frighten ghost away. During the game, the user would collect various item, such as mana potions, or health hearts to restore all power/health lost by a small portion, so that the character continues to fight off ghost that she encounters. Part of the difficulty of the game arose from the vast darkness the character is, and what she will do if she encounters a ghost coming close to her. Also, since there is no map through the game, the user must figure out a way to battle all the ghost before reaching the end.

This specification is essentially a re-implementation of the framework “Walk”. As with the original walk game, the game specified here is single user only. The specification, of course, open ended, in that a dark setting, items, and enemy ghosts are added.

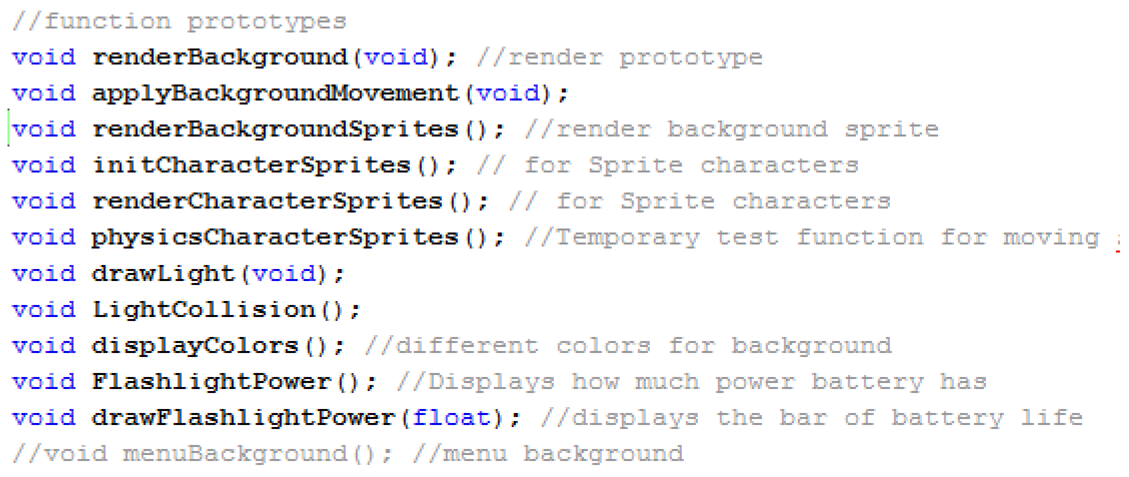
Figure 4

2.2 Product Functions

- Start Menu
- Pause Screen
- Game Over Menu
- Help/Settings Screen
- Player Health
- Player Lives
- Player Mana
- Key Functionality
- Background Wrapper
- Music added when collecting Mana/Health items
- Power Up
- Miscellaneous (add-ons)

2.3 User Classes and Characteristics

The user classes and functions will be inherited from three different .cpp files that each individual member from the team stored their functions to compile onto the game.cpp file. The game.cpp acts as the main file, which will grab all the used functions to compile from the Makefile ./game. For instance:



```
//function prototypes
void renderBackground(void); //render prototype
void applyBackgroundMovement(void);
void renderBackgroundSprites(); //render background sprite
void initCharacterSprites(); // for Sprite characters
void renderCharacterSprites(); // for Sprite characters
void physicsCharacterSprites(); //Temporary test function for moving
void drawLight(void);
void LightCollision();
void displayColors(); //different colors for background
void FlashlightPower(); //Displays how much power battery has
void drawFlashlightPower(float); //displays the bar of battery life
//void menuBackground(); //menu background
```

Figure 5

2.4 Operating Environment

The main component of Beyond The Light project is the Linux operating system that manages all of the hardware resources associated with the local and remote software and hardware. All team members modified and performed all of the sources on Linux.

2.5 Design and Implementation Constraints

The challenges in the development of the product included the inability to write in OpenGL from the beginning of the course. Furthermore, we continue learning OpenGL from trial-and-error, and from other sources online as a guide to write the program functions. Another challenge the team faced was gathering a reassuring idea of what the game functions should perform, and thus delaying our individual sources from being written into a functional program. Currently, we are under development of the game, adding features on it every week to deliver on time.

2.6 User Documentation

Along with the product, the user help options will be written to help students and programmers understand the usage of the developed game system. The user help options would follow with key system features and operations; step-by-step instructions for using the system.

2.7 Assumptions and Dependencies

The project would build on Linux operating systems. In this regard, necessary inspirations could be obtained by analyzing related systems such as X Window System, Apple Mac OS, Microsoft Windows, Ubuntu, and NetBeans. In particular, the design and implementation approach of the Ubuntu operating system could be helpful to draw a clear guideline for developing the intended game/project.

3. External Interface Requirements

3.1 User Interfaces

All interaction with the user is via a command line/key interface. Once the game starts, the user is prompted by the main menu to start by using keys to go to the next stage of the game. If the user enters the wrong key, the system will not understand the command and will not do anything. Yet, if the user uses the correct key, the system will understand the command and perform the user’s key input.

Figure 6 shows an example of how the menu for the user will input using key strokes.

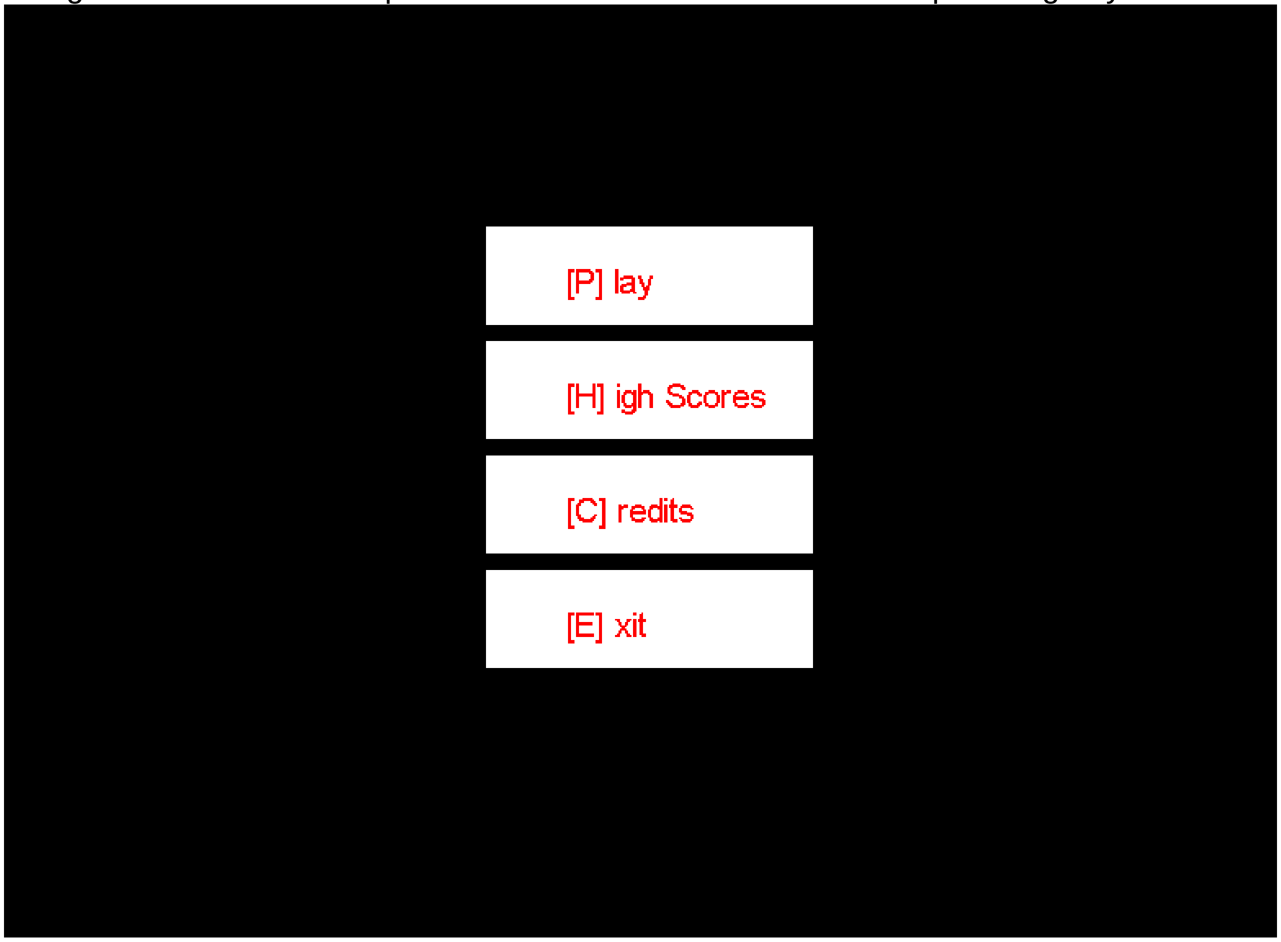


Figure 6

3.2 Software Interfaces

Beyond The Light is to be developed under the Linux operating systems using Ubuntu's remote server. The system shall be provided with one type of user interface. OpenGL is the software interface heavily use for the development of the program, without use of core commands from glut.

3.3 Communications Interfaces

Beyond The Light is performed through Sleipnir and/or Ubuntu's remote server to properly run the programs' framework and other components. Incoming data consists of updates from the server regarding the user's score, health points, and mana. Outgoing data consist of user's information (such as name), confirmation to start the game, and how many items they collected. The game will interact by displaying the user's outcome of the game, such as if the user runs low on game life, they get a game over, or if they collect many items or terminate many ghost (randomized objects) they earn a high score or the highest score. Furthermore, the game will call for a database system that stores the user's information.

4. Misc.

Some of the challenges faced when writing the program was setting the background and the objects. In the following, we have the layers of the background, and continue working on adding tiles for the character to jump to give other alternatives of the key inputs the user can perform while playing the game.

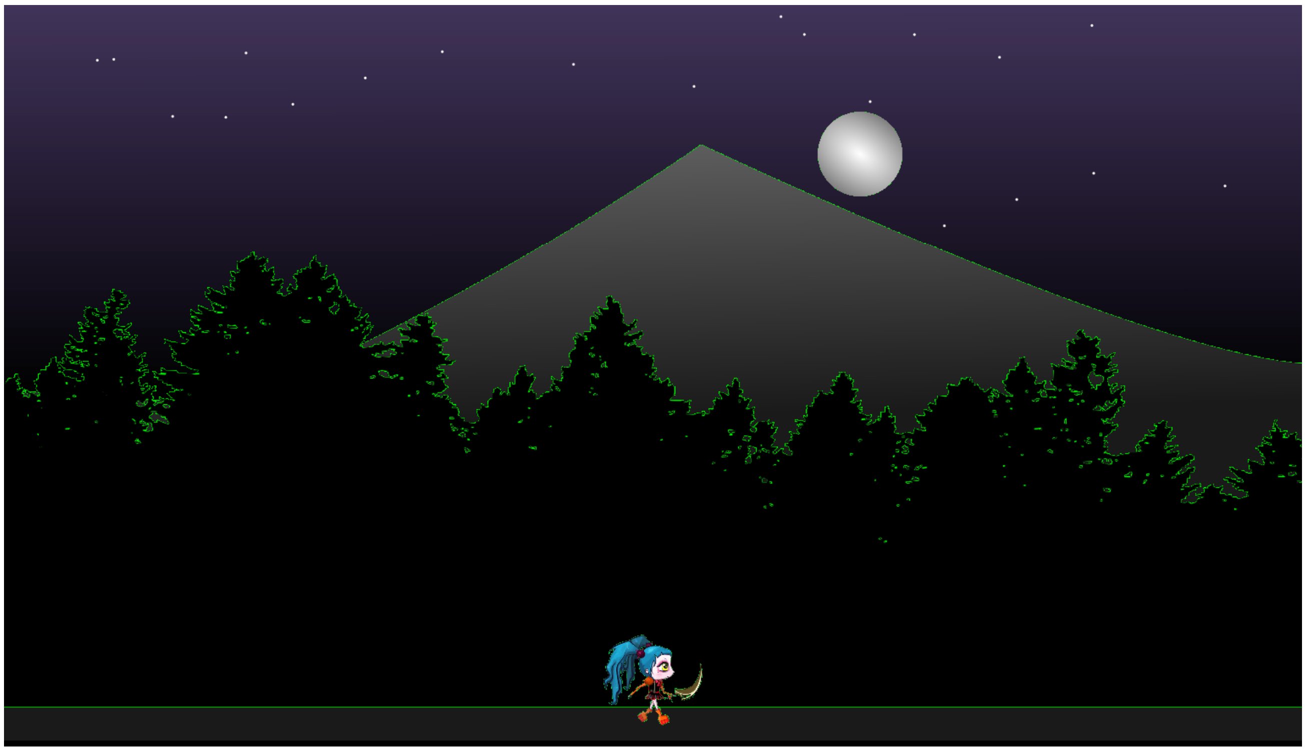


Figure 7

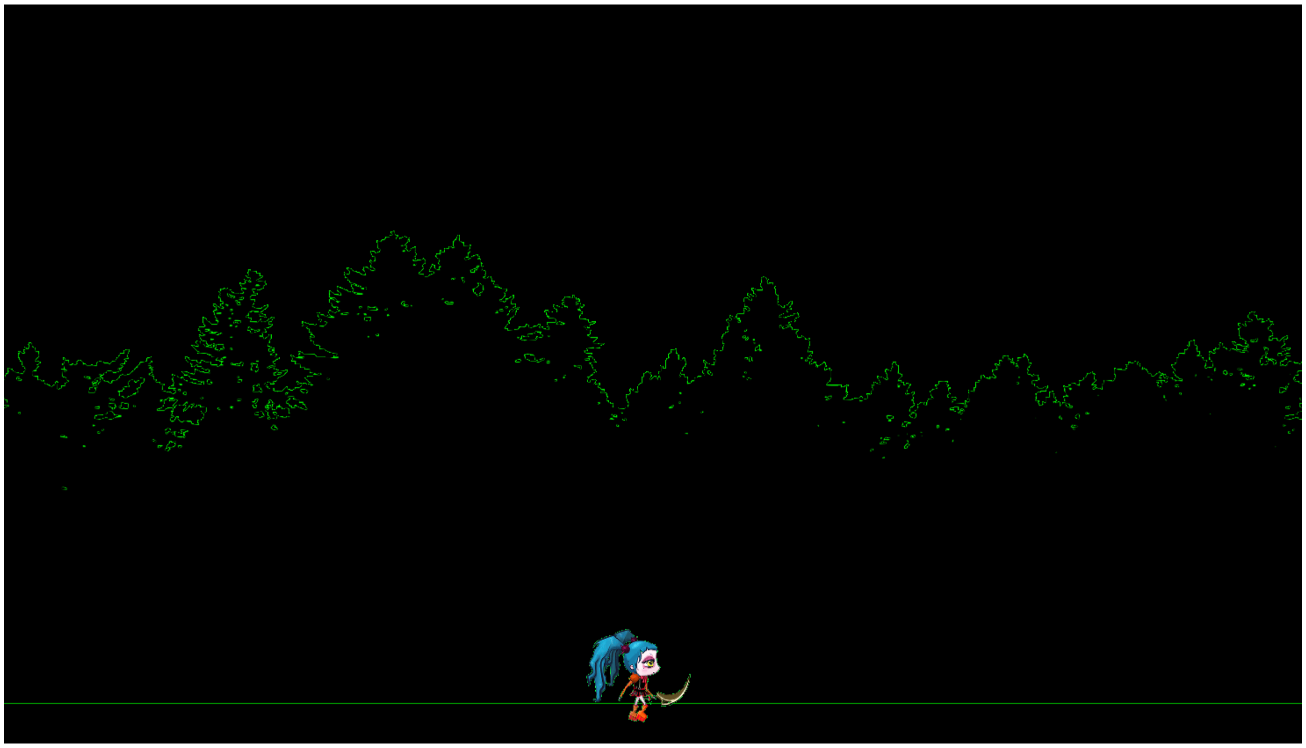


Figure 8