Game Design With Unreal Engine and C++

Zoe Castle, Will Coffey, Emilia Germain, Zoe Leger, Alexa Pace, and Dawson Wright

Appalachian Regional Commission/Oak Ridge National Laboratory Math-Science-Technology Institute 2020

Introduction

Game development has a wide variety of uses, such as simulations, entertainment, and problem-solving development in a wide variety of fields. Some examples of these fields are physics, neuroscience, and mathematics. Using simulations could help train someone who is learning a dangerous job, such as being a pilot. With these simulations, the pilots gain experience while not actually being in danger. Entertainment is one of the main uses for game development. With game development, you can make puzzle games that are fun but also help build problem-solving skills. The end goal of our project was to have a C++ based game that uses inheritance, functions and object oriented programming.

Background

A Brief History of Video Games

- First video game was OXO by A.S. Douglas in 1952.
- First video game that could be played on multiple computer installations was *Spacewar!* by Steve Russel.
- Most early games were made using raw machine code
- First commercially successful video game was Pong by Atari in 1972
- C++ and Java are the most common languages for modern games and are used alongside gaming engines like Unreal Engine and Unity.

Unreal Engine -

- Published by Epic Games in 1998.
- Program that is widely used to develop games and models.
- Advanced way to create 3D experiences through first person shooters, platformers, RPG's, fighting games, third person games, and many other genres.

C++ -

- Programming language developed by Bjarne Strouped in 1979.
- Statically typed and object oriented (meaning it uses different types of classes) as well as using data type (int, float, etc.), functions, inheritance, etc.
- Widely used with Visual Studio.

GitHub -

- Founded in 2008.
- Widely used to save multiple versions of game/code and go back to previous versions.
- Lets users share and collaborate on a project.

Materials and Methods

Materials: Computer, Unreal Engine, Visual Studio, Github

Want to Create a New Project?	**Unreal Engine**	**Github**	**Visual Studio**
First Steps	• Create a new Unreal C++ file • Choose a template or blank map • Go to the content folder, right click, click show in explorer, this is your projects location.	• Go to your github • Click add repository, give it a title, a description, choose public, add git ignore. • Copy url (found when you click the green box that says code) • Go to the unreal project location, open a git bash, type **git clone** paste url.	• A new file/project should open when you create a C++ file in Unreal.
Creating a Map	• Click on landscape under modes • Choose a size for your landscape then click create • Under the modes tab you find tools that will help you create a landscape • Play around with the tools until you get the result you want.		
Creating an Object	• Inside of your project right click and choose blueprint class • Choose actor and double click on the item when it shows up. • Choose add component to give the actor a shape, and add a type of collision • Choose the event graph tab • Here you can code your object.		• Once you create your object in Unreal, you can edit the code in Visual Studio instead of the Event Graph.
Saving	• Click save current at the top. • This will save the project to your device.	• Upload updated code to your repository by opening your file. • Opening a Git bash. Type **git add .** • Once it loads type **git commit** • Type **i**, add a comment, press esc, type **:x**, enter • Type **git push** • This saves your project to your repo so your can download it from other devices.	• Click file • Click save • This will save the project to your device

Results

The goal of the project was to create different individual levels with an item that required coding for it to do a specific function. In total, six levels were created that each had their own goal and interactive item, shown in Figures 3-8. Below is one example of one of the levels that was made.

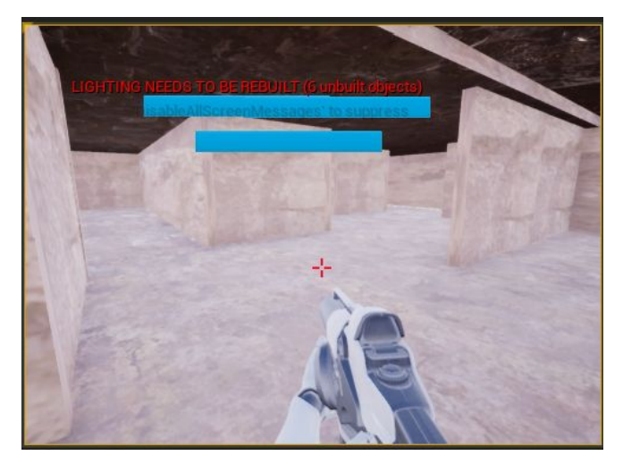
Figure 1

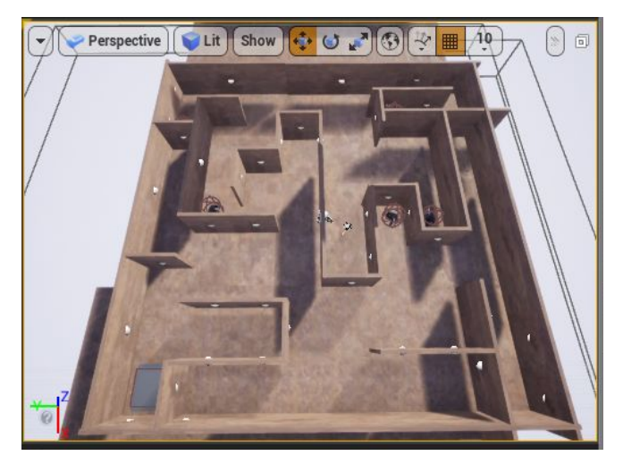
Figure 2

In this level you are simply trying to move from one side of the game to the other. To do so the player must travel through a maze with twists and turns while encountering enemies to defeat along the way. Upon reaching the other side of the maze the player encounters a platform that serves as an elevator to transport them upwards to the next level of the maze. Figures 1 and 2 depict the second level that consists of a maze slightly harder than the one the player just completed before it. While creating this game, difficulty was encountered when creating the C++ blueprint code for the platform to make it functional. At first attempt, the player was unable to be moved from the first level to the second because they could not move through the floor. For the platform to be operational the floor of the level was made in separate pieces rather than a whole one with the piece the platform needed to go through having no collision so that the player could be moved through it.

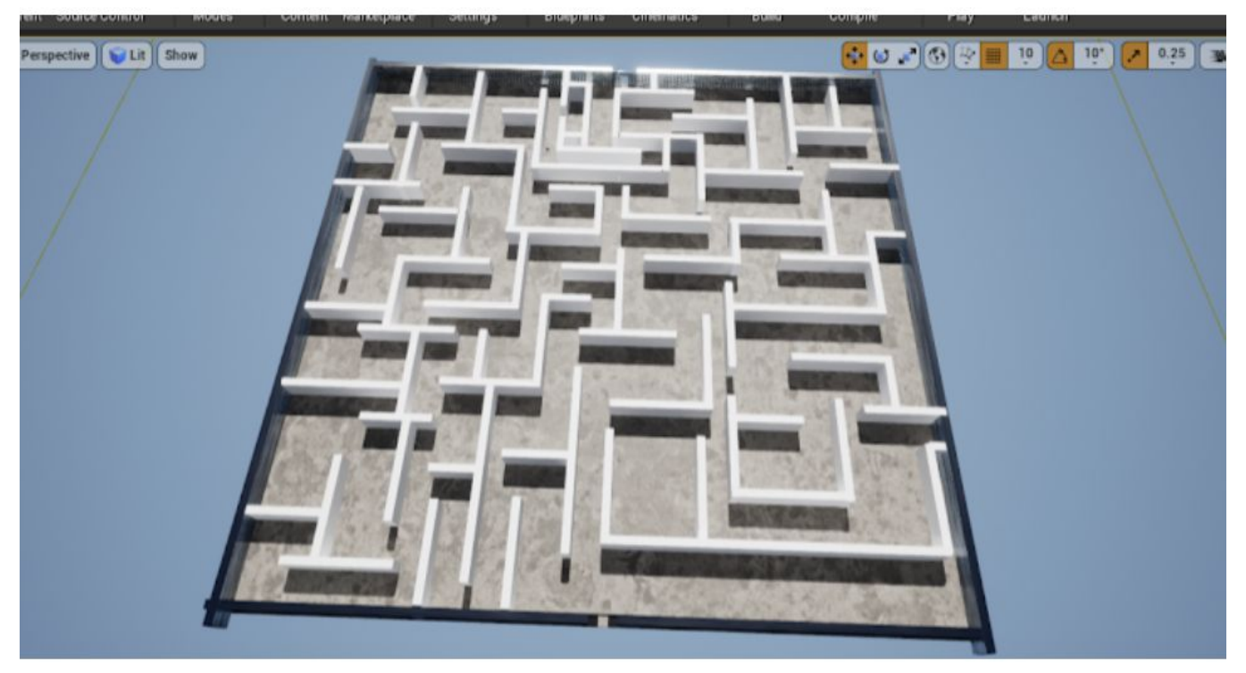
Figure. 3 Alexa Pace's Level

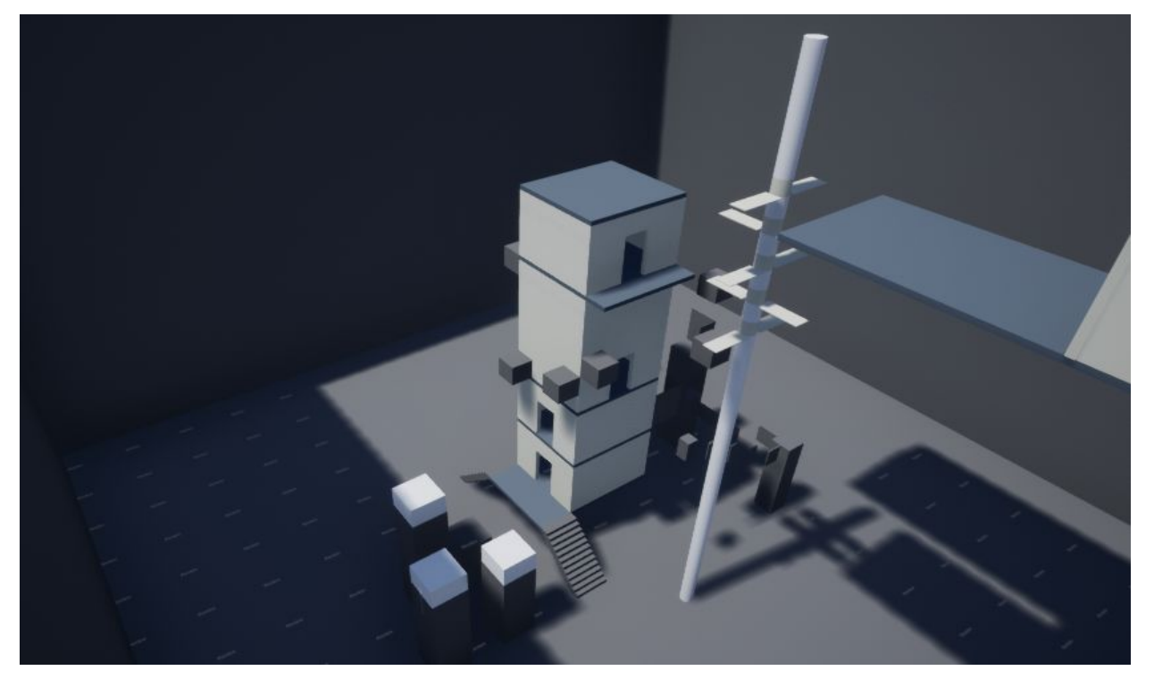
Figure. 4 Dawson Wright's Level

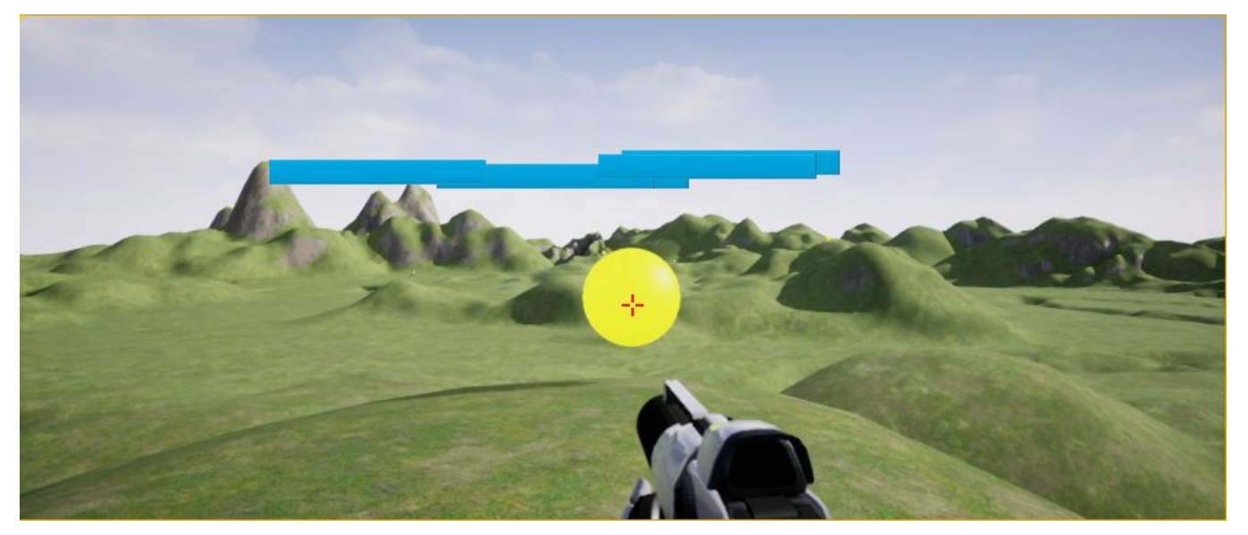
Figure. 5 Emilia Germain's Level

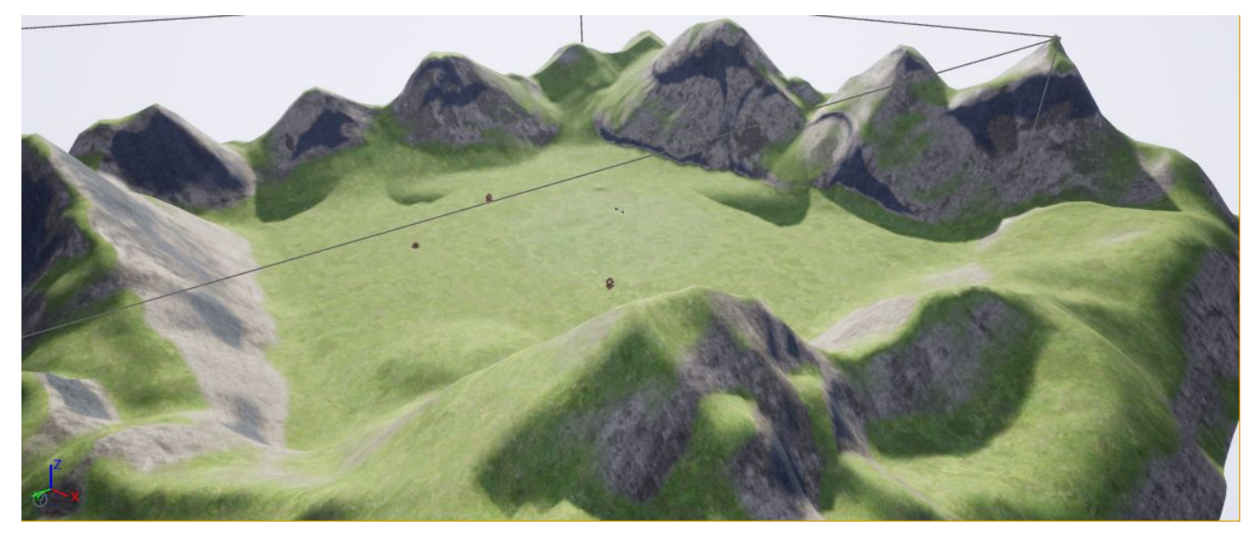
Figure. 6 Zoe Leger's Level

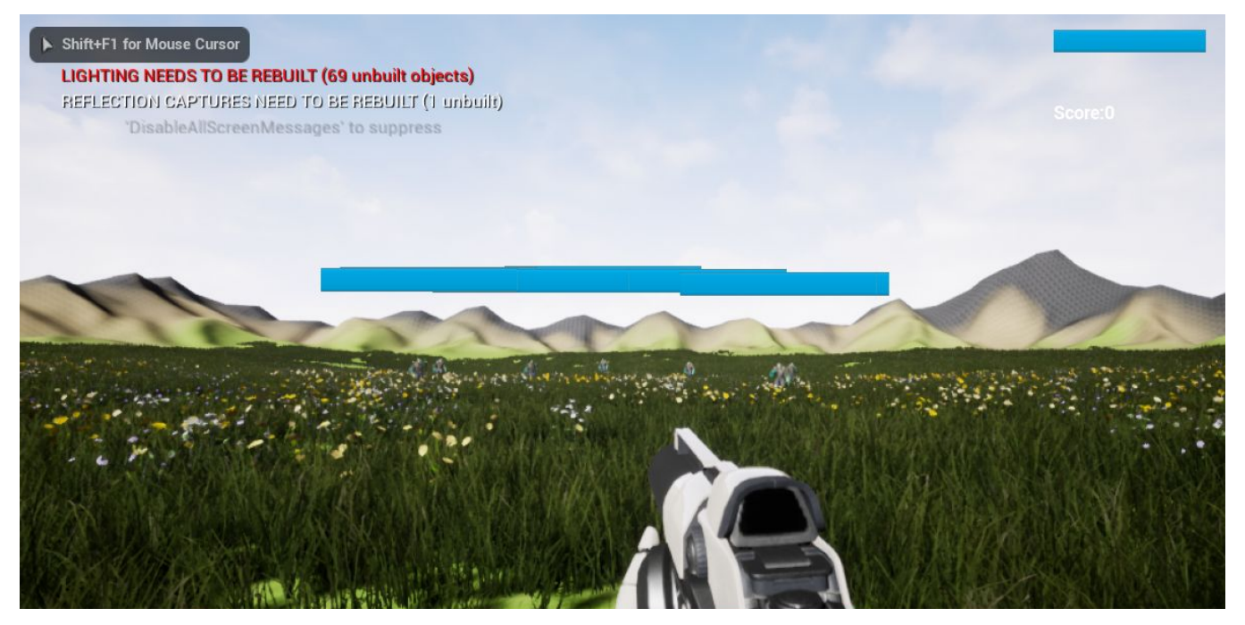
Figure. 7 Will Coffey's Level

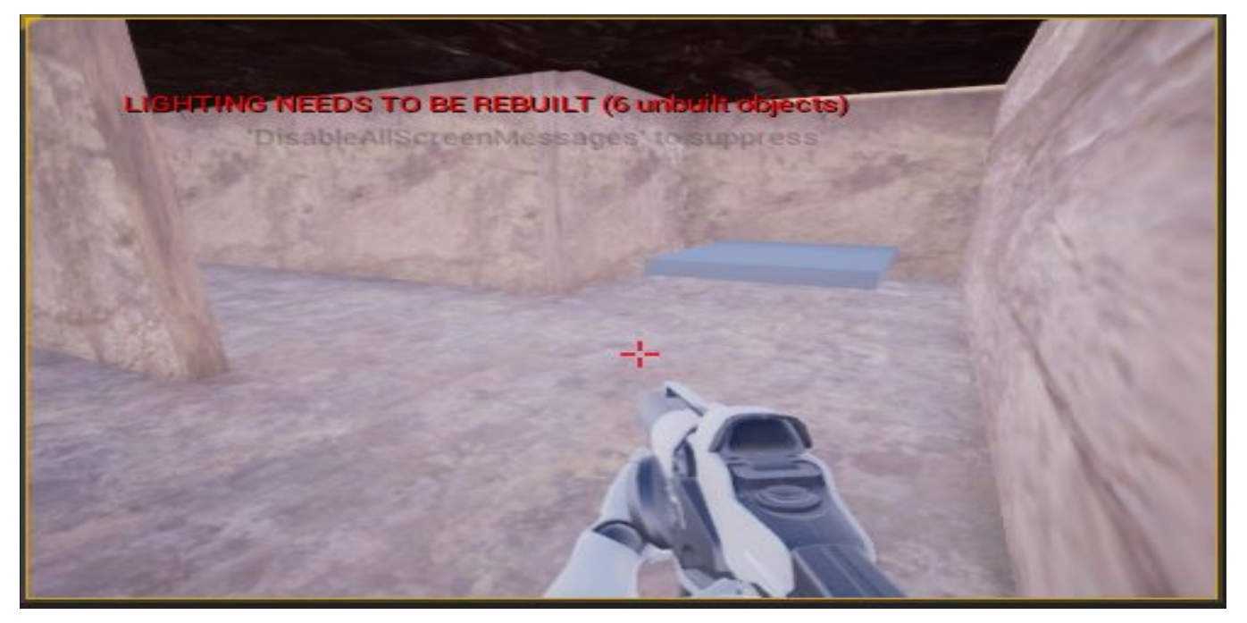
Figure. 8 Zoe Castle's Level

Conclusions

The goal of this project was to learn the basics of game design in order to create multiple different levels of a game. In addition to this, real-world applications of game design were learned. Coding and game development can be used to create successful games. However, it can also create different real-life simulations for a variety of applications. For example, instead of having real-life training, people in the military could train by having a sophisticated simulation so that the risk of them being injured would not be present. Another example, is using Unreal Engine to create a simulation of a possibly high-risk experiment before conducting them in real life. This saves money and time. Also, AI technology could help us in our everyday lives by letting us learn from our mistakes and learn from different experiences. It can be as simple as a chess-playing computer, or it could be a self-driving car. One common AI that many people use is the Roomba that vacuums your house by itself.

Acknowledgements

Thank you to Skyler Valdez, Stephanie Biddle, Gary Cipinko, Chris Nelson, Jennifer Tyrell, and Kayla Canario. We appreciate the opportunity provided to us by Sandia National Laboratories, Oak Ridge Associated Universities and the Appalachian Regional Commission.