James Caleb Way

Bachelor of Science in Computer Science

Dr. Sean Hayes, Project Advisor

Expected Graduation date, December 2023

Problem Statement

Video games lack replay value and to solve this issue procedurally generated worlds, multiple starting classes, and numerous secrets should be implemented into video games.

The ideal state of a video game is one that has a high replay value. If it has a high replay value, then the player will continue to come back to the game. Additionally, it allows the player to experience something new with each playthrough. However, many games are generic and do not offer diversity in gameplay which leads to games getting played only once. Video games need to have a high replay value so that people will continue to play them even after they finish them.

Replay value is very important. Many people will play a video game once and then never play it again. So, by providing the player with reasons to replay the game, they will be able to get more hours of enjoyment through multiple playthroughs. A few ways to encourage players to replay a game is with a vast number of achievements, secrets, alternate story lines, side quest, starting character classes, etc. All of these add to the replay value. Additionally, games like *Minecraft* and *Valheim* have replay value because they use procedural world generation. And unlike other games which have predefined worlds, worlds that are procedurally generated prevent the player from knowing where items, structures, bosses, etc. are so the player will need to search for them. Furthermore, games like *Detroit: Become Human*—who’s story has over 40 different endings—encourages the player to replay the game multiple times to unlock all the endings. Likewise, *Elden Ring* and *Dark Souls* have multiple starting character classes for the player to choose from. This encourages the player to replay the game multiple times to experience the unique playstyle of each class.

To address the problem, I propose the implementation of a video game with procedural world generation, multiple starting classes, and secrets for the player to find. This will allow players to have a new experience with each playthrough and allow them to see aspects of the game that they missed on their first playthrough. Also, it will create more diversity in how the player plays the game and would inspire them to play in a new way.

In conclusion, the lack of replay value of video games can be challenged with procedural world generation, multiple starting classes for the player to choose from, and numerous secrets for the player to find.

Project Description

This project will be a melee combat video game for Windows. To achieve this, it will use the wave collapse function to procedurally generate worlds. The game will be single player and open world where the player can explore in whatever way they choose. Because of procedural generating the bosses will be randomly spread out so the player can fight the bosses in whatever order they would like. Enemies, plants, and merchants will be randomly distributed throughout the world but will be confined to certain regions. Also, there will be multiple starting character classes for the player to choose from. Each class will have different stats and different perk weapons that will deal more damage. Enemies that the player defeats will drop different items of random quality based on the enemy’s tier. The player will be able to loot these items. Additionally, defeated enemies will give the player experience points that can be used to upgrade the character’s skills. Also, the player will be able to harvest materials from plants and chest that they can then use to craft different items like health potions. The player will be able to visit merchants to buy weapons, armor, and crafting materials. Furthermore, checkpoints will be randomly spread across the world. If the player rest at the checkpoint, then their health and health potions will be restored but resting will respawn enemies. The last checkpoint that the player visited will be set to the player’s respawn point for if an enemy kills them. Finally, the player will have a map of the land that they have explored, marking the player’s current location, merchant locations, discovered boss locations, and visited checkpoints. The player will be able to fast travel between visited checkpoints by using the map.

Proposed Implementation Language(s)

C++ and Unreal Engine Blueprints.

Libraries, Packages, Development Kits, etc. to be used in in the proposed implementation languages(s)

Wave Collapse Function

Additional Software/Equipment Needed

Unreal Engine 5, Visual Studio, and Blender.

Personal Motivation

I love video games. I have always enjoyed designing levels for games through their built-in level creator, but I have always wanted to make my own game. Additionally, I want to work at a video game studio and throughout my college career I have learned many things. However, I haven’t learned how these things relate to developing a video game. What's more, many game studios use Unreal Engine to build video games. So, by building this game in Unreal Engine 5 I will gain experience with how the engine works and will have an understanding of how games are built. All of this will help prepare me for entering the game development industry.

Outline of Future Research Efforts

Future research efforts will include but are not limited to downloading and installing Unreal Engine 5, Visual Studio, and Blender. As well as research into the Wave Collapse function for world generation and how to use it within UE5. Research on how to make an optimized game in UE5. And additional research in level design to allow the player to progress through the game with a good balance of challenge and accomplishment.

Schedule

* Download and install Unreal Engine 5, Visual Studio, and Blender (11/21/2022)
* Start research of Wave Collapse function (11/21/2022)
* Setup a testing environment in UE5 and start implementing and testing Wave Collapse (11/21/2022)
* Submit proposal and requirements documents (11/21/2022)
* Finish wave collapse testing (1/31/2023)
* Create project environment and implement working wave collapse (2/8/2023)
* Begin implementing combat, weapon, and looting systems (2/8/2023)
* Begin implementing enemy AI (2/8/2023)
* Finish implementing and testing desired world generation (2/30/2023)
* Begin implementing foraging, looting, and crafting systems (2/30/2023)
* Begin implementing spawn groups and zones (3/15/2023)
* Finish up all work (4/30/2023)