Kronos

A Virtual Reality Roguelike

Reily Stanford
Union City, Tennessee
rstanfo1@utm.edu

Enrique Tejeda
Martin, Tennessee
enrgteje@ut.utm.edu

ABSTRACT

Kronos is a virtual reality video game with roguelike elements in which time is falling apart at the seams. It falls on the protagonist, Kronos, to restore the timeline to its chronological order. The player is capable of manipulating time to assist with handling any challenges faced. The game is a first-person shooter where you are traveling between a mismatch of time periods in order to get back to the present to restore order. Kronos is built using Unreal Engine 5 and the plugins it provides. Some of the behaviors of the enemies within the game, the item effects, and the weapon mechanics are created using the C++ language. There are a variety of floors, each having its own time period theme.

Author Keywords

Authors' choice; of terms; separated; by semicolons; include commas, within terms only; this section is required.

1. INTRODUCTION

Kronos is a virtual reality game that is set in the first person. The art style is similar to DOOM (1993) in that the enemies are pixel-art 2D sprites that always face you. The game is a shooter where you are tasked to bring the timeline to its original state. Someone has disrupted the chronological order of events leading Kronos, the protagonist, to fix the timeline by any means possible. There are three levels that each correspond to a time period where you can easily tell that time has been altered. This is apparent when you see enemies that do not match the current time period the level is set in.

Mechanics

A primary mechanic that the game has is procedural generation of level layouts. The game offers three methods of movement: teleportation, head-based locomotion, and hand-based locomotion. With teleportation, you move by pointing your hand to a specific location and pressing the movement button. The other option, locomotion, moves like a standard video game in which you hold the movement button to move in your desired direction. Each level is concluded by defeating a boss enemy that is stronger than the enemies you have been fighting on the current floor. There will be multiple weapons, items that affect your player's stats, and a variety of enemies on each floor.

2. TECHNICAL SPECIFICATIONS

The project is built with Unreal Engine 5 [5] and we used Unreal's Blueprints, a visual programming language, to create the logic of the game. For complex functionality that would be too visually complex to be built with Blueprints, we create with C++ [NOTE: WE HAVEN'T MESSED WITH C++ AS OF YET]. We use a plugin called Procedural Dungeon Plugin [2] that places the rooms into a playable layout. To plan our roadmap we used Milanote [7]. The assets we have in our project come from Humble Bundle [3], itch.io [6], and Mixamo [1]. As for source control, we used Git LFS [4], which is used for large files on github, and the built-in source control for Unreal Engine.

REFERENCES

1. Adobe. Mixamo. https://www.mixamo.com/, 2022.
2. BenPyton. Procedural dungeon plugin. https://github.com/BenPyton/ProceduralDungeon, 2019.
3. Humble Bundle. Sfx and music for your games. https://www.humblebundle.com/software/sfx-and-music-for-your-games-software, 2022.
4. GitHub. Git large file storage. https://git-lfs.github.com/, 2014.
5. Epic Games Inc. The Unreal Engine. https://www.unrealengine.com/en-US, 2022.
6. itch.io. itch.io. https://itch.io/, 2022.
7. Milanote. Milanote. https://milanote.com, 2019.