GitHub Repository: https://github.com/liammshaw/RealSeniorProject
YouTube Link: https://www.youtube.com/watch?v=WBn-WRpi7KY

Introduction

While there is no shortage of cozy, comforting sandbox games on today's market, finding a cooperative, two-player, puzzle-based game is a scarcity. Further, finding one of these games which operates on one machine, locally, is even rarer. While cooperative experiences are often readily available over the internet, our game introduces another layer of collaboration with real, face-to-face interaction.

The main motivation for this project stems from a mutual lifelong love for video games, specifically sandbox games. These types of games, such as Mojang's Minecraft, ConcernedApe's Stardew Valley, and many more, have consistently provided us comfort in stressful times throughout our lives. Now, at the culmination of our undergraduate experience, the curriculum we have experienced throughout these past four years has provided us with the knowledge and resources that allows us to provide this same sense of comfort for others. Our game aims to encourage collaboration between two players through solving puzzles and minigames. In addition, our game seeks to inspire players to find creative and innovative solutions to the challenges they face. The success of our project can be evaluated primarily through verifying planned functionality of the game, as well as through surveying player enjoyment and satisfaction through user playtesting.

In our game, we have succeeded in creating a cozy, comforting two-player game in which players can sit side-by-side and further intensify the level of connection within the multiplayer experience. With the ever-increasing possibilities over the internet, oftentimes the real player you are interacting with gets lost behind the screen. Our game breaks these barriers and reinstates the reminder of how important human connection is.

Background

Our game was developed using the Unity game engine. Prior to this project, neither of us had had any experience with the engine, but both had a visceral interest in learning the software. Unity is a very powerful game development engine which has many advantages. One advantage is the readily available online resources for the software; there are countless video tutorials, help forums, and documentation, and code libraries for the software which speed up development immensely. In addition, Unity has great support for 2D games as well as 3D games, and features better cross-platform integration than other popular game engines. Some disadvantages of using Unity include a lack of support for demanding graphics, the cost of the Unity license, and how memory-intensive the software is, which makes it less accessible for less powerful machines.
In addition, our scripts were developed in C#, which is an object-oriented, component-based programming language. Because Unity is also component-focused, this programming language best suited the needs of our project.

**Related Work**

Our project draws inspiration from an abundance of works, each pertaining to different aspects of gameplay. The most influential of all is the adventure game It Takes Two by Hazelight Studios, which also emphasizes the theme of collaboration between two players by solving puzzles in order to revitalize their marriage. The main difference between our game and It Takes Two is that our game functions on a two-dimensional playspace featuring pixel art while the world of It Takes Two is three-dimensional. In addition, our game does not feature a romantic relationship between the two players, allowing for a more expansive primary audience. In addition, gameplay is similar to that of Humongous Entertainment games like Freddi Fish and Pajama Sam in the sense that the player must gather tools and use them in creative ways in order to achieve a certain goal and progress through the game. However, our game involves puzzles that require two people to solve rather than one.

Our game's movement and perspective is similar to that of Stardew Valley, by ConcernedApe, and Pokémon, by the Pokémon Company. In addition, the pixel art style of these games (particularly less recent Pokémon versions like Platinum, Soulsilver, and Black) largely mimics the art style for our game. Finally, the main inspiration for the art style and thematics of our game is drawn from several Studio Ghibli films, most notably Castle in the Sky, Howl's Moving Castle, and Spirited Away (among many others). Each of these sources provides inspiration for an essential component of our game design, resulting in a unique and enriching player experience.

**Design**

Our project is a cooperative two-player, tile-based video game in which players need to collect objects and work together to complete levels. The players can control their sprites with one player using the W, A, S, and D keys, and the other player using the Left, Right, Up, and Down arrow keys. When the game begins, the players spawn in a forest setting, brimming with life. Green grass, tall trees, and stumps litter the meadow, and at the top is a large, overgrown, rusty robot that is embedded into a rocky cliffside. The robot is missing its left arm and is unresponsive. A door can be seen at the base of the robot. The players must work together to find the missing limb and repair the robot in order to complete the game. Near the bottom of the screen is a fiery purple spirit that advises the players to head to the beach, which resides in the play zone to the southern direction of the robot.
On the beach, the players find sparkling blue water and golden sand. To their left, there is a bridge leading to a docked boat. Upon entering the boat, they find a scuba suit and a SONAR device. One player may put on the scuba suit while the other calibrates the SONAR device. The player wearing the scuba suit enters the water while the other player continues to configure the SONAR device. The player on the SONAR device calibrates it using a "simon says" style pattern. Each successful round will spawn a whirlpool in the ocean for the player in the scuba suit to interact with, which will complete the round. After five rounds, the players are able to retrieve the missing robot arm from the bottom of the sea.
Figure 2: Beach scene with adjacent bridge.

Figure 3: Boat and ocean scene.
Upon returning to the forest clearing, the players find that the robot's arm is repaired and the door at its base has opened. Upon entering the door, they find that they must repair the robot's faulty circuitry by rerouting the wires. The players must enter the circuit on two avenues. Then, the players must remain within adequate proximity of each other in order to turn on the six lights throughout the circuit. After doing so, the players have completed the game.
The puzzles have been designed so they require two people working in harmony in order to solve them. Players must use their problem-solving and critical thinking skills to retrieve the robot's arm and repair its circuitry. Our target audience is pairs of people who are interested in improving their teamwork and collaboration skills. This includes, but is not limited to: romantic partners, pairs of friends, siblings, or even coworkers and colleagues.

**Implementation**

For our project, we decided to use the Unity game engine for C# development. We decided to use C# for development because that is the programming language in which the primary inspiration for our project, Stardew Valley, was developed with. Our development process was driven by a variety of game development tutorials on an as-needed basis.
From our research, we found that Object Oriented Programming is almost synonymous with game development. To make life easy for ourselves we decided to take an object-oriented approach. The figure above outlines how we broke down our classes:

The **Game Controller** class acts as the overall manager for our game. It has two instances of the Player Controller class (player one and player two), an enum GameState, booleans that manage puzzles, and Audio Sources. The game states can be broken down into three states, Free-roam, Dialog, and Paused. In the free-roam state, the players can navigate the world as they wish. When the game switches to the dialog state, the game will pause and play whatever dialog was prompted. Lastly, the paused state will only occur when the players change scenes. There are two booleans RobotComplete and PuzzleComplete. Those will become true when the respective puzzle is completed. Finally, the GameController manages the sounds and music of the game using the Audio Sources.

The **Player Controller** class manages everything associated with each playable character. It has an instance of the **Character** class which stores all the essential data associated with each character and takes in keyboard inputs to determine what action needs to be taken.
The figure above describes some of the helper classes. The **Dialog** class stores the text for any instance of dialog. The **Dialog Manager** class displays the dialog when the game state is updated. The **Portal** class switches the player between scenes. The **Game Layers** class manages all the different layers of our game to make future additions easier.
The last part of our implementation consists of the item classes. All Item classes inherit the interactable interface. The **Robot**, **ScubaSuit**, **Whirlpool**, **SonarDevice**, **CircuitLight**, all implement the interact() function found in the interactable interface. The SonarDevice also acts as the controller for the SimonSays minigame. When the player interacts with the sonar device a panel will open up and the player must finish the game in order to progress in the game.

**Analysis / Verification**

This quarter, our goal was to complete two whole puzzles: the boat puzzle and the robot puzzle, as well as add detail to the scenes through visual assets and audio clips. Because of the work that we put into making our code reusable and scalable throughout the first phase of senior project development, we were able to complete this goal with ease. On top of finishing the two puzzles, we were able to flush out the different scenes with sounds, animations and a lot more details than our first iteration.

This quarter we were also to provide our peers with multiple rounds of playtesting. Our goal for the playtesting was to get feedback on the artwork, gameplay and overall feel of our game and
we got meaningful and constructive feedback after each one. After the first playtest, the overall feedback was that our game felt unfinished and confusing. To tackle this we added a NPC in every scene with some dialog to provide hints to the players, the purple flame sprite in each scene. We also added appropriate detail assets to each scene, resulting in an overall more interesting and immersive gameplay experience. We added animated sea grass and rocks in the water and an illusion of deeper water, more trees and plants to the forest, more detail to the sand and water in the beach scene, and more detail to the floor tiles. After the second playtest, we got much more positive feedback. Our peers said that the levels felt much more alive and had a much more complete feel. They also said that the NPCs were a nice addition in the case where they were unsure what they were intended to do.

Through player feedback and playtesting, we ensured that we succeeded in creating a cozy, comforting video game with satisfying audio, visuals, and gameplay. Each successive round of user playtesting resulted in a higher quality game than the previous.

**Future Work**

Given more time, we would like to expand this game to include more levels, more complex problem-solving, and improved graphics and animations. Our initial planning stages included far more puzzles, levels, and gameplay. However, due to the nature of the quarter system, and with only 20 weeks of development time from essentially no base knowledge, it was not feasible to include all of these amenities in our work. However, we plan to continue to expand and improve this project far beyond the scope of this course.

Specific works include adding more puzzles to our initial level. One idea we had initially discussed was a forest scene with a sleeping creature, which could be awoken with the smell of food. Additionally, we discussed the setting of our game as a mystic island whose summit is an active volcano. The end goal of the game would be to free the dragon who had been trapped in the volcano for centuries.

**Conclusion**

Out of all the projects that we have taken part in during our time at Cal Poly, this is the most fun and rewarding experience we have had thus far. We will have been able to experience the entire game development cycle as well as using the Unity game engine as a development environment. We developed a scalable, reusable C# codebase, created our own stylized artwork, and provided a sufficient proof of concept for our final product upon the conclusion of spring quarter. Throughout this development period, there were many highs and many lows, but we are proud of our end product and are confident that we have built a steady foundation for development next quarter.
Works Cited


Hayao Miyazaki (Director). (2002). Spirited Away [Film]. Studio Ghibli.

Appendix

Appendix A: Gameplay Beat Sheet

Legend:
Rectangle - Stage
Diamond - Action
Oval - Dialogue Prompt

Spawn

Navigate to Robot

Robot

Explore Map
Missing Arm - Puzzle

Talk to Sprite at the bottom of the spawn

Navigate to Ship through the Beach

One player enters the water
One player controls Sonar Device
Player interacts with whirlpools after they spawn
Other player plays simonsays to spawn whirlpool

Robot arm disappears; talk to sprite
Electricity - Puzzle

Robot opens

Enter left side of Robot Maze  Enter right side of Robot Maze

Interact with lights to complete puzzle players must be close

Power Turns on Exit Robot
Appendix B: Meadow and Beach Scenes from Senior Project I