6-13-2019

The Labyrinth

Derek Nakamura

Marko Trapani

Liam Walker

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 10, 2019

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Derek Nakamura
Marko Trapani
Liam Walker

ENTITLED

The Labyrinth

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF COMPUTER SCIENCE AND ENGINEERING

Thesis Advisor

Department Chair

The Labyrinth

by

Derek Nakamura
Marko Trapani
Liam Walker

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 13, 2019

The Labyrinth

Derek Nakamura
Marko Trapani
Liam Walker

Department of Computer Engineering
Santa Clara University
June 13, 2019

ABSTRACT

The Labyrinth is a puzzle video game meant to be experienced with a virtual reality headset. It aims to differ from typical puzzle games by being simultaneously fun, immersive, and mentally stimulating. The game consists of three unique level types, with room for further development to create more complex versions of said levels. In this document, we discuss how we developed the game environment to effectively engage the player in enjoyable and engaging problem-solving challenges. The game has been rigidly tested and play-testers were surveyed on their experiences, in order to improve the game.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Since the 1980s, video games have substantially grown in popularity as an entertainment medium. From violent shooting games to mindless phone games, the vast majority of games do not stimulate the critical mind. People often sink many hours into games like Grand Theft Auto or Angry Birds due to their thrilling or addictive nature. A meta study by Stetson University found that games such as these are prone to producing undesirable behavior in players, such as aggressiveness. The mobile game market somewhat mitigates this problem by providing a variety of popular puzzle game options. But mobile games are limited mechanically and are forced into a small two-dimensional screen. Experiences offered by non-VR puzzle games on game consoles or computers are further restricted by the devices supported for interacting with the puzzles, such as game controllers or a mouse and keyboard. On top of this, games which explicitly attempt to develop problem solving skills too often end up feeling like boring IQ tests. Its clear that these media are insufficient to optimally engage and develop a players problem solving abilities for a three-dimensional world.

1.2 Solution

Virtual reality has been growing in popularity among technology enthusiasts and common consumers within the past couple of years. It introduces a new form of immersion for simulated environments, like those found in video games, because it puts the user into a three-dimensional space for exploration. Consider how the Rubiks Cube is a significant upgrade in difficulty from a pencil-and-paper maze due to its additional dimension. Like the Rubiks cube, our solution, The Labyrinth, will leverage the additional dimension of virtual reality to enable a more challenging and immersive puzzle game experience. The Labyrinth will present the player with levels of increasing difficulty. A VR world can offer the surreal experience of solving the puzzles in a uniquely hands-on way; for example, piecing together floating objects or teleporting through space. The Labyrinth uses the Oculus Rift, a popular virtual reality headset and controller system. The Rift provides the player with a relatively easy setup and user-friendly controls. Furthermore, the

in-game graphics will be aesthetically pleasing to promote user immersion. By immersing the user in the virtual world of the puzzle and providing ways to solve problems more enjoyably in virtual reality, The Labyrinth will promote problem solving skills while being fun for users.

Chapter 2

Requirements

The requirements describe what the system does at a high-level and limits its scope. Requirements can be broken down into two sections, functional and non-functional. The functional requirements explain what the system will do, and the non-functional requirements explain how the functional requirements are fulfilled. Constraints describe limitations within the system.

2.1 Functional Requirements

- Critical
  - Start screen with following options:
    - Play Game
    - Level Select
    - Settings Menu
    - Quit
  - Ability to pause the game and access Settings menu
  - Levels populated by objects and entities
  - 3 types of puzzles
  - Increase in difficulty as the game progresses
  - Allow for custom settings with respect to sound and graphics
- Recommended
  - Heads up display
    - Mini-Map

    - Hints system
    - Narration (text or voice)
    - Different puzzle per level
        * Smaller puzzles within each level
- Suggested
    - Progresses linearly through a story

2.2 Non-Functional

- Critical
    - Be difficult enough to be challenging, but not too difficult that it is frustrating
    - Be fun to play
    - Be mentally stimulating
    - Be smooth to play with respect to frame rate and bugs
- Recommended
    - Be designed with aesthetics in mind
- Suggested
    - Have the feel of a finished game
    - Have good sound effects

2.3 Design Constraints

- Movement should not make users nauseous
- Must run on Oculus Rift
- Must be made using the Unreal Engine
- Frame rate (FPS) must be sufficiently high

Chapter 3

Use Cases

3.1 Use Case Diagram

The use case diagram illustrates how an actor would typically use our system, and the different actions they may take. The only type of user is a player. The player will have a number of actions that they can take to complete different levels in the game.

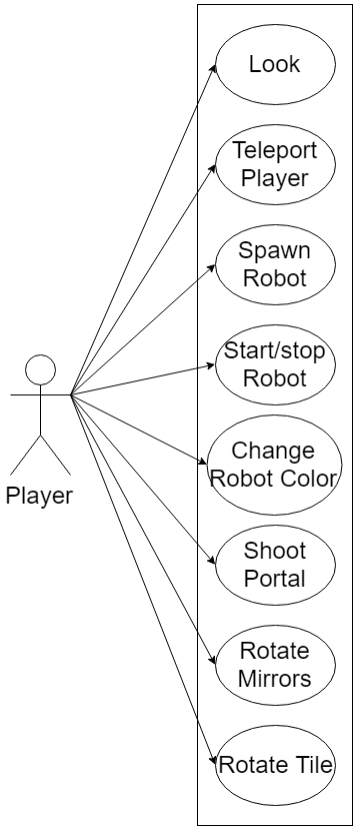

Figure 3.1: Use Case Diagram

3.1.1 Player

- Look
  - Goal: To allow the player to visually explore the level before making decisions.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
  - Post-Conditions: The player is finished with the game or exits the game
  - Exceptions: None
- Teleport Player
  - Goal: To allow the player to move throughout the level.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
  - Post-Conditions: The player is finished with the game or exits the game
  - Exceptions: None
- Spawn Robot
  - Goal: To allow the player to restart level three by spawning or respawning the robot element.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The plyaer is playing level three
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None
- Start/Stop Robot

  - Goal: To allow players to control the pace of the game. By allowing the player to start and stop the robot, he can spend longer choosing how to solve the puzzle.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The player is playing level three
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None

- Change Robot Color
  - Goal: To give the player extra challenge in puzzle solving.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The player is playing level three
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None

- Shoot Portal
  - Goal: Allow the player to choose the path of the robot element.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The player is playing level three
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None

- Rotate Mirrors

  - Goal: To give players control of the laser element.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The player is playing level two
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None

- Rotate Tile
  - Goal: To give players control of the puzzle path.
  - Actors: Player
  - Preconditions:
    * The player has set up the Oculus Rift
    * The player is wearing the Oculus Rift and has opened the game
    * The player is playing level one
  - Post-Conditions: The player is finished with the level or exits the game
  - Exceptions: None

Chapter 4

Activity Diagrams

The activity diagrams illustrate how users will interact with each of the completed level.

4.1 Level 1

In level 1, the player is placed into a room with circuit tiles placed along the walls. The tiles on the wall can be rotated 90 degrees in either direction. The goal of the level is to rotate the tiles in the proper sequence so that the start tile has an unbroken circuit connecting it to all the end tiles.

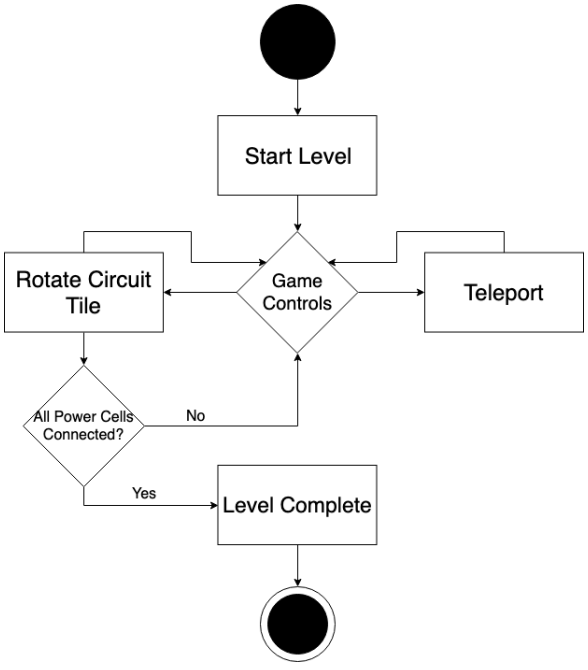

Figure 4.1: Level 1 Activity Diagram

4.2 Level 2

In level 2 the player is in a maze. There are three stationary laser emitters, red, blue and, green, placed next to the player's start area. There are a number of mirrors placed throughout the level, as well as reflective walls. The mirrors reflect the lasers from the emitters and can be rotated by the player. There are also red, green, and blue laser sensors placed throughout the level. When a laser sensor is hit by its matching color laser, it lights up a large orb in the sky above it. The goal of the level is to simultaneously light up the red, green, and blue orbs in the sky.

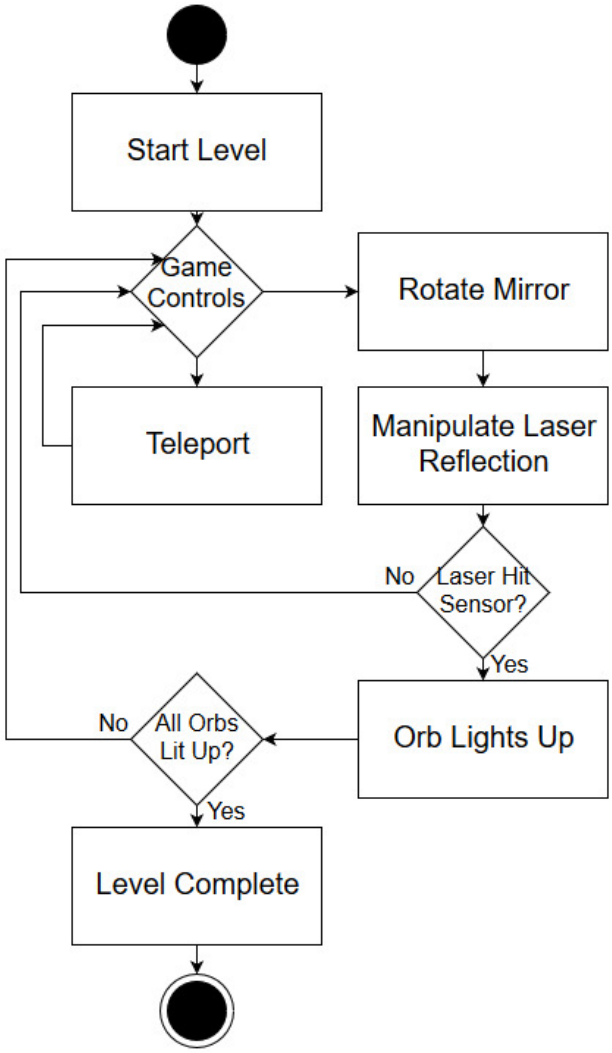

Figure 4.2: Level 2 Activity Diagram

4.3 Level 3

In level 3, the player is place in a room with hoops and cubes. The player can spawn a robot, start or stop the robot's movement, change the robot's color, or place two different colored portal openings on the cubes. The robot will move in a straight line once spawned and can pass through portals. If the robot passes through a hoop of mismatching color, the robot is destroyed. If the robot matches the color of the hoop, then it can pass through. The goal of the level is for the player to get the robot to move from its starting point to its end-zone.

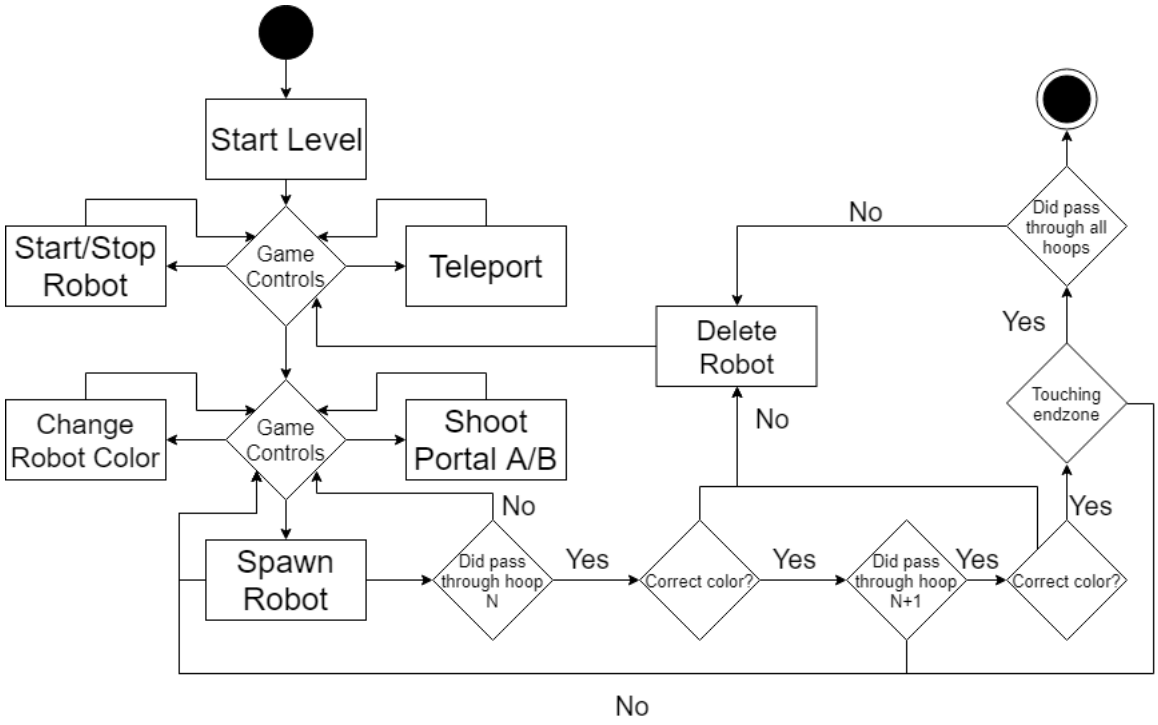

Figure 4.3: Level 3 Activity Diagram

Chapter 5

Architecture Diagram

The architecture of game design within unreal engine is event-based and object-oriented. We will provide an overview of classes and their functions in our architecture diagram. Should new functions or classes need to be implemented, we have allowed some room for new classes with abstraction.

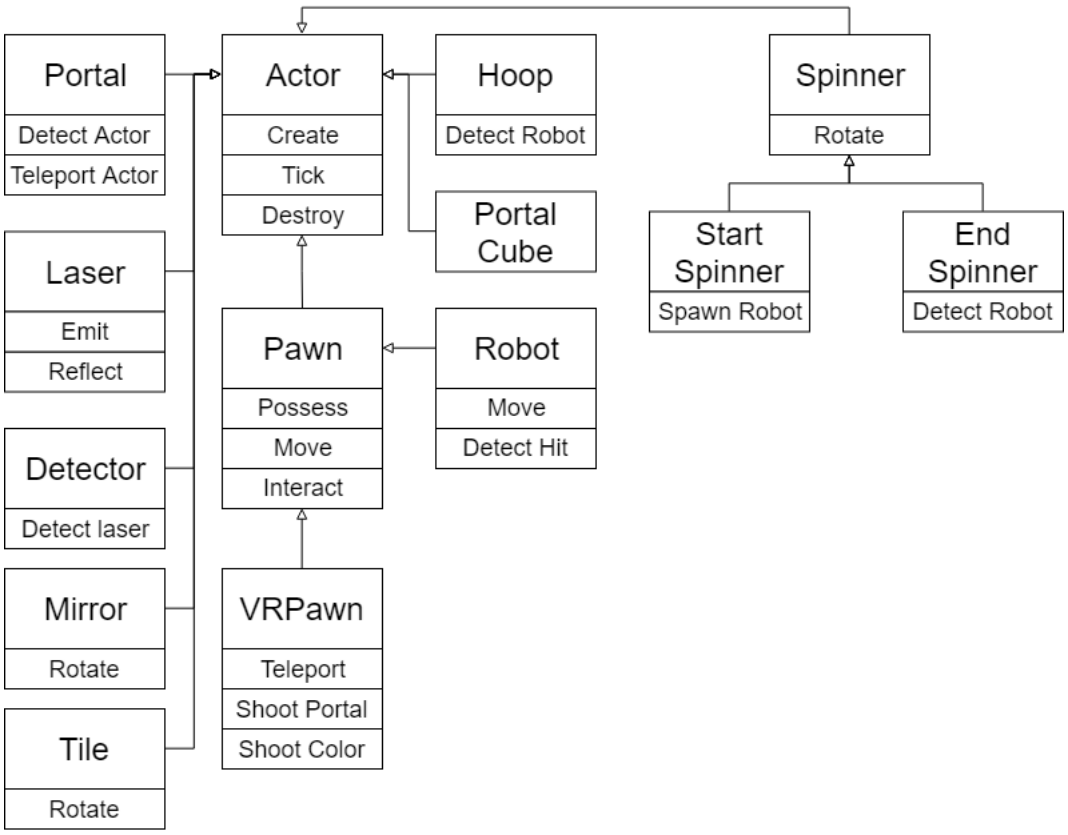

Figure 5.1: Architectural Diagram (Object Oriented)

We didn't use too many layers of abstraction for this design because each of our classes have very unique behavior. For example, a laser doesn't have much in common with a robot in terms of model, movement, or environment detection.

There are some elements that have more than one level of abstraction: the VRPawn, Robot, Start Spinner, and End Spinner. The VRPawn and Robot both inherit from Pawn, the class in Unreal Engine which specifies that the object moves and can interact with the environment. VRPawn is the player class, so it fits that behavior. Robot is the object in level three which moves around the level from start to finish and interacts with walls and portals. The Start Spinner and End Spinner both have the same model, spinning behavior, and detection of the robot.

Chapter 6

Technologies Used

6.1 Oculus Rift

The Oculus Rift is a Virtual Reality Headset and comes with ergonomic controllers which are used to experience and interface with the game (6). Any game in virtual reality requires a virtual reality headset kit and a set of controller to play.

6.2 Unreal Engine 4

Unreal Engine 4 is a game engine (1). A game engine gives the developer a set of base tools as well as a platform upon which to build the game. The game engine that will be used to develop and test the game is Unreal Engine 4, which has built-in VR support.

6.2.1 Blueprint Visual Scripting

Blueprint is Unreal Engine's gameplay scripting system (7). It is used to design all of the objects in the game and define each of their attributes and functionalities.

6.2.2 Software Packages

Developers provide software packages online which can be used to aid development in Unreal Engine. We bought useful packages with pre-allocated grant capital.

Chapter 7

Design

7.1 Level Design

Creating levels in Unreal Engine 4 is similar across different types of games. We all had some experience with the engine, and we had some familiarity with level design. The game consists of three distinct level types. As the game progresses, new features are introduced to increase the difficulty of the game; features such as pressure plates, where the user is required to stand on a piece of ground which will perform an action, lasers, where the user is required to move around the lasers or be forced to restart the level, and robots, where the user is required to avoid detection and or be forced to restart the level.

7.2 Class Design

Classes which require programmable behavior in Unreal Engine 4 require them to inherit from the actor class. An actor in Unreal Engine 4 is a parent class that allows the object to be placed within the game engine. All of our classes within the engine have specific functions which are described in our architectural diagram. We used an object-oriented design to build our game as is traditional to video game creation.

Chapter 8

Design Rationale

8.1 Maze-Oriented Design

Originally, we planned to design the game as a maze over-world which the user would traverse, solving sub-puzzles along the way. However, mid-way through development we found that the traversal was not fun or meaningful to the experience; we also ran into many technical challenges which slowed our development progress.

8.2 Puzzle-Oriented Design

With our new design, we instead focused on fleshing out a smaller number of puzzles. We stuck to three sub-puzzles and expanded them into their own levels. We also chose to remove the maze structure. We chose these design decisions in order to focus on what made our game good: the sub-puzzles. Even though we abandoned the maze structure, we decided to keep the old name, because we still felt it fit the essence of our game.

8.3 Oculus Rift

Our game will use the Oculus Rift for the player interface because it requires less time and effort to set up. It also has a more intuitive interface than other popular VR headsets (5). For example, the controllers offer a traditional joystick and many more buttons than other headset kits. We chose the Oculus Rift because the player may want less of a drastic change when moving into virtual reality from traditional gaming.

8.4 Unreal Engine 4

This will be used as the game engine because it has VR support (1). While Unity is another popular game engine for independent developers, we chose Unreal Engine 4 since our group collectively had some familiarity with the engine before starting development.

8.4.1 Blueprint Visual Scripting

Unreal Engine only uses Blueprints for VR development. It is relatively simple to use because it uses nodes to represent each object or gameplay element. The nodes can be connected to one another to design the basis for the game. Each node can also be modified to change the attributes of each objects.

Chapter 9

Testing

This will explain testing of the system to find bugs and methods for improving the system.

9.1 Completed Testing

These are the testing procedures that we have currently finished.

9.1.1 Alpha Testing

Testing of individual modules/features. After each module was completed, tests should were run by both the developer of the feature and another developer. Developers also did a complete run through of their levels to see if they can be completed and to see if they found any major or noticeable bugs.

9.1.2 Weekly Design Reviews

Weekly design reviews were held with other developers to check on progress and future planning for development. Similarly, weekly design reviews were also held with out advisor for the same purposes. These were useful in keeping development on track, deciding if a problem was holding back progress, and if more work needed to be done to get back on track.

9.2 Future Testing

These are the testing procedures that we plan on completing in the future.

9.2.1 Beta Testing

A select few volunteers will test out levels to see if they find any bugs within the game. This will also be done to check if any of the levels make users feel nauseous.

9.2.2 Player Surveys

After playing, volunteers will be asked questions about their experience to see their feelings about difficulty of levels, if they had fun, and any feedback for improvements.

Chapter 10

Development Timeline

Tasks are not labelled by person. We chose not to assign tasks because we had an idea of when each task should be finished, but did not have an accurate idea of how long each task would take. This design decision allowed for flexibility, but did require more planning before development. We were able to roughly follow this timeline, but there were a number of setbacks that held back a lot of our early development, i.e. the shift in design philosophy. We also had a couple of incomplete requirements, like a start menu, level select, settings menu, and a quit option, because of setbacks during development. The following development timeline was created using GanttProject: a free project management software (4).

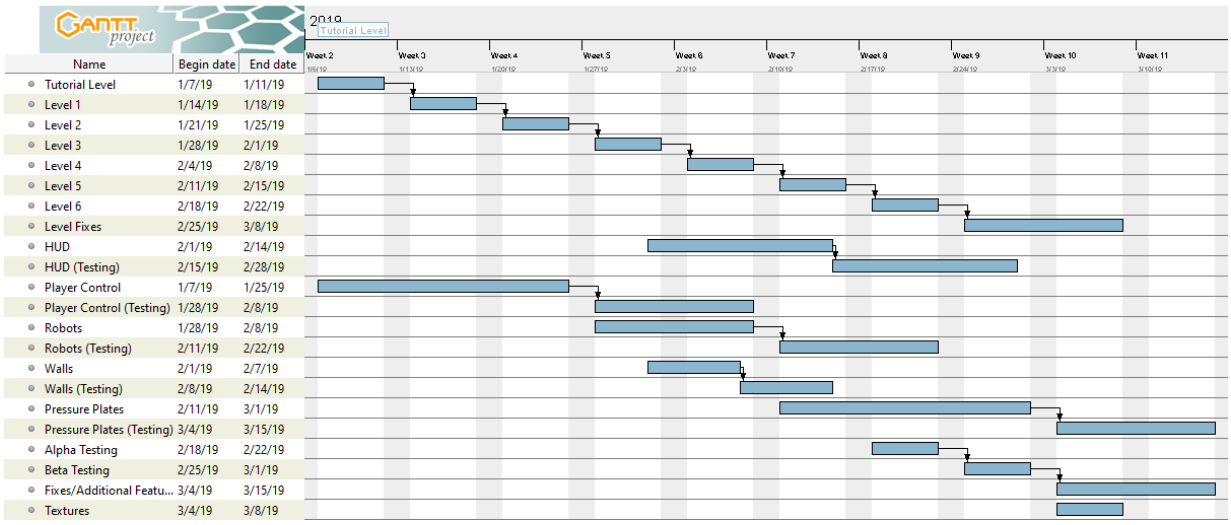

Figure 10.1: The Labyrinth Overview

Chapter 11

Risk Analysis

The risk table shows the risks and consequences involved with the completion of the system. The table also shows the probability that a risk will occur along with a severity ranking for the consequences. The probability and severity are multiplied to approximate an impact measurement. Each risk has mitigation strategies listed for reducing the likelihood of a risk occurring along with the impact of its consequence.

Risk	Consequence	Probability	Severity	Impact	Mitigation
Time	System not completed on time	.5	8	4	Set deadlines Set priorities on tasks Attempt to accurately predict time to develop features
Bugs	Parts or all of game does not work properly	.4	10	4	Test early and often
Game is not enjoyable to play	Nobody will want to play the game or people will give up on the game quickly	.3	10	3	Emphasize user testing and implement user requests to make the game more enjoyable
Game progress is either too hard or too easy	Users will feel frustrated if it is too hard and unfulfilled if it is too easy	.6	5	3	User testing and feedback on their perceived level of difficulty Implement skippable levels and hint system
Game is unpolished	The game looks and feels like it is unfinished although functional requirements have been satisfied	.3	8	2.4	Test frequently Polish high priority features regularly

Figure 11.1: Risk Analysis Table

The mitigation strategies listed were used during our development process to ensure that few of the consequences occurred and the system was completed and functional by the deadline. The first two risks, time and bugs, were encountered during development. We were able to use our mitigation strategies to make up for any setbacks in development and to complete our system on time. The last three risks we predicted were very likely to occur with our original design philosophy. In order to circumvent these issues we shifted our design philosophy from a maze design to smaller levels with a single well-thought out puzzle within each. The shift ensured that our game was completed on time and was well-polished for users to play.

Chapter 12

Societal Issues

12.1 Ethical

Video games and ethics haven't had the best relationship historically, at least in the public eye. Between video game addiction and the media-alleged link between violence and video games, some may be inclined to believe that the video game industry as a whole is an ethical net negative. However, video games aren't going away any time soon, and we believe that video games like The Labyrinth, which are designed to be mentally stimulating and non-addictive, can in fact provide a positive and ethically-sound experience for the player.

12.2 Social

The Labyrinth could have a positive impact on society because it will help users to improve different aspects of their thinking. As a VR puzzle game, users are challenged to improve their critical thinking skills as well as improve their spatial awareness. Critical thinking is an important skill that people bring into their jobs and other parts of society. Integrating better critical thinking into society enables people to become better at solving problems and furthers the likelihood that they will make good decisions for themselves and others around them. The negative social impact that The Labyrinth could have is a reduction in users' interaction with others in society. If someone is spends a lot of time playing a game like the Labyrinth, then they are probably engaging in little to no interaction with other human beings, as it is a single player game. Even if someone is nearby watching them play, the Oculus headset covers a player's eyes and prevents face-to-face interaction between players and other human beings.

12.3 Political

While the political ramifications of virtual reality are yet unclear, the VR industry is of legitimate concern for the future of public life. If VR tech is to become as or even more popular and pervasive than that of cell phones, certain regulatory steps will likely demand to be taken. While seemingly unlikely, it would be quite undesirable if young people were to get so sucked into virtual reality worlds that they prioritize it over their lives outside of VR. Perhaps

virtual tech will need to be age restricted or usage limited to a certain number of hours a day to incentivize people to maintain healthy levels of engagement in real life.

12.4 Economic

The VR industry is growing rapidly and promises to be a very large part of the gaming industry within a few years (8). Where that will lead is unclear, but it also seems likely that augmented reality tech will also be further integrated into daily life for practical and non-entertainment purposes as well.

12.5 Health and Safety

As far as we know, the health concerns for VR use are relatively minimal. However, in rare cases, seizures have been observed in some after exposure to certain flashing light patters or prolonged usage of VR. On the other hand, it is somewhat common for users to experience dizziness or nausea after extended use. In order to avoid symptoms like this, it is best to limit one's use of VR to shorter periods of use and to take the necessary precautions not to bump into other people or objects while playing (10).

12.6 Manufacturability

Since this is software, manufacturability refers to the ease of deployment and the ease of distribution of the software. As of now, our game is on GitHub, so it is downloadable and playable only if distributed privately by the developers. Our game development engine, Unreal Engine 4, makes it very easy to convert our game into a playable application. There is an operation within the engine to export the game as an executable. These two aspects of our design makes our game very "manufacturable". In our case, we have a long-term plan of putting this game on the "Steam Marketplace," a central hub for downloading video games. This game would have to be approved by Steam, which demands a variety of requirements that the game does not yet have, so there is still work to be done to achieve this.

12.7 Sustainability

This is a sustainable product because it is software. As software, the game can be updated at any time through patches that users can download to update their version of the game or to fix any bugs. Updates could be used to add more content to the game, like new levels, which can provide more entertainment and use time for users.

12.8 Environmental Impact

Our project does not have much negative environmental impact. All of our environmental impact comes from the cost of running our game on a computer. Our game is somewhat CPU-and-GPU-intensive, so the power output of the user's

computer will slightly increase. We believe this is a negligible cost to the environment since it is only a few degrees Fahrenheit increase on the user's computer.

12.9 Usability

While our game does not have specific documentation to go along with it, the user interface was designed to be intuitive. Since we developed our game with the user in mind and with a user-friendly interface as a goal, we can say that our project has a high usability. For example, within level one, the player has the goal of connecting pipes to orient the maze from start to finish. Each pipe along the way lights up to show that it is connected, so the user only has to connect a single link in the puzzle to understand how the entire puzzle is solved. All of our levels were built with this in mind. We designed our game to be easy to figure out rather than simply giving the user instructions on exactly how to solve each puzzle. One downside to our usability, however, is the fact that our game is played in virtual reality. Since this is a new technology, many users will have trouble simply looking or moving around within the game. We believe this learning curve will decrease over time when more people have become familiar with virtual reality technology.

12.10 Lifelong Learning

During this project, we learned some lessons that we will take with us for our professional career and for the rest of our life. The most important of these lessons learned was from our shift in design philosophy. The basic premise of the "sunken cost fallacy" is that as people, were reluctant to give up what we have in order to get something better, even if we can see that its better. We had invested a lot in the old version of the game, so it was hard to give up. We could all see that changing was going to benefit our project much more. The lesson we learned here was that even though we had a lot of time spent on one path, we must not hold on to the past when we know it's wrong. Part of the trouble with developing in our game engine was that even though we all had some experience using the engine, it was still a relatively new technology for us. Especially in virtual reality, which none of us had experience in. The entire process was a learning experience of how to be a game developer, which requires looking with a different perspective than than application development for school courses. At first, our strategy was to look at online tutorials for how to program certain elements or behaviors. But sometimes, we could follow the tutorial exactly and it would simply not work: it could have been a different engine version or an interaction with a slightly different class. This problem would keep popping up. What we learned was that sometimes, you have to program from scratch based on your intuitions and looking up information on the specific library functions. Only after you get completely stuck can you consult the guide.

12.11 Compassion

Although compassion was not our sole motivation to create this game, we do believe that compassion had a vital role to play in our project. From the start, we wanted a game that was going to be useful to its players rather than simply fun. In our initial problem statement, we contrasted our game to the very popular games of our time, Angry Birds, which is mindlessly simple, and Grand Theft Auto, which is mindlessly violent. We wanted to move away from those games in terms of their usefulness to the players while keeping the fun spirit of the game that makes players engaged. There has been research that video games improve cognitive skills of the user (9). We also wanted to make our game fun so that the user can actually enjoy the benefits that a mentally stimulating video game can provide, so we set the goal of our game being fun as one of our top requirements.

Chapter 13

Obstacles Encountered

The obstacles and setbacks that we encountered during our development process are listed below.

13.1 Teleporters

The first obstacle that we encountered were teleporters. Initially, we planned to have teleporters placed throughout the labyrinth that players could use to teleport to different rooms. The teleporters we used were downloaded from the Unreal Marketplace, but they were designed for non-VR games. These did not work in VR because the player does not have an actual collision box in VR. We spent a long amount of development time trying to implement said collision box and debug the teleporters, but we eventually gave up and decided to get rid of the teleporter idea. It was not until the last few weeks of development that we built our own working version of the teleporters using our own design. The functioning version of teleportation worked only for objects, but was successful for the purposes of a level. Our initial approach was to follow a tutorial step by step from the internet and then use our own skills to debug anything that was not the same as the internet resource (3). We learned that this was not the best development practice to use and we came up with a much better practice that we used later in development.

13.2 Learning a New Technology

A recurring issue was that we were not experienced in all of the nuances of Unreal Engine 4 (1). While we had some experience developing in Unreal, we did not know everything about it and we would often not know exactly how to create certain objects or we might attempt to implement an object in a way that was not completely correct. Again, we started off with the development practice of looking up online tutorials and trying to follow them exactly, implementing our own changes afterwards. We often found that certain aspects of the tutorial implementation would not work well with our systems, so we ended up spending a lot of time to debug our version to get them to work. More often than not, we found that this would not result in a functioning version of what we wanted.

13.3 Unreal's Version Control

Unreal has its own version control built into the engine, so we assumed that we could use it easily. However, we found out that there were a lot of issues with synchronizing our versions across our different systems using Unreal's version control (1). To work around this issue, we used Git as our method for version control. Git is the most well-known and reliable form of version control so it was an easy choice for solving this issue (2).

13.4 Throwing Mechanic

Initially we designed throwable balls that could be used to change the color of a robot if they were hit. We thought that a throwing mechanic in VR would be enjoyable for players. However, during testing it became clear that it was very difficult to throw a ball to hit a robot as it was moving in the air. We decided to scrap this idea and change it to a point and click mechanic to change the color of a robot.

13.5 Shift in Design Philosophy

We originally planned to have a labyrinth design for the overall layout of the game. This had the major flaw of being very repetitive for the player because they would have to memorize the directions that they took to get to the end of each section of the labyrinth. They would be met with some smaller puzzles along the way, but we realized that this would become rather tedious and did not seem enjoyable when we were designing the puzzles. Rather than having to design a lot of small incoherent puzzles to scatter throughout a maze, we decided to completely shift our design philosophy for the game. Our new design philosophy was centered around three separate levels. Each of these levels would have their own well-thought out puzzle that would feel polished.

Chapter 14

Conclusion

We completed three levels of The Labyrinth. These levels show that there is a functioning game for users to play. We learned a couple of lessons from our setbacks. First, we learned that sometimes its better to scrap an idea if it is holding back development, like teleporters, Unreal's version control, and the throwing mechanic. Second, we learned that its better to build something from scratch first, and then consult internet resources for debugging purposes, like with the teleporters. And lastly, we learned not to be afraid to change our game's design philosophy when things weren't working well. These were important lessons to learn, not just for this project, but for any future projects that we may be apart of. We also learned that VR games are much more immersive for players than traditional games designed for 2-dimensional screens. Players moving around in a 3-dimensional space and using their hands to control objects is more engaging and provides another level of immersion in gameplay. Our game is currently functional and challenges players to solve puzzles in a 3-dimensional space. Our game could be improved further by implementing many of the features that we did not finish, like the start menu, level select, settings menu, and quit option. More challenging levels could also be implemented by developing new puzzles or combining the currently implemented mechanics together.

References

[1] Unreal Engine 4.

https://www.unrealengine.com/en-US/

[2] GitHub.

https://github.com/features/project-management/

[3] Unreal Engine Forums.

https://forums.unrealengine.com/community/community-content-tools-and-tutorials/3103-tutorial-creating-a-teleporter

[4] GanttProject. Project planning technology.

https://www.ganttproject.biz

[5] Jon Martindale. Oculus Rift vs HTC Vive. December 14, 2018.

https://www.digitaltrends.com/virtual-reality/oculus-rift-vs-htc-vive/

[6] Oculus Rift.

https://www.oculus.com/rift/oui-csl-rift-games=mages-tale

[7] Blueprint Visual Scripting.

https://docs.unrealengine.com/en-US/Engine/Blueprints/index.html

[8] VR Industry Growth.

https://www.gamesindustry.biz/articles/2018-12-06-ar-vr-spending-to-jump-69-percent-in-2019-idc

[9] Nauert PhD, R. (2018). Video Games Can Help Boost Social, Memory and Cognitive Skills. Psych Central. Retrieved on June 3, 2019, from

https://psychcentral.com/news/2013/11/26/video-games-help-boost-social-memory-cognitive-skills/62537.html

[10] VR Health and Safety Guide.

https://www.classvr.com/health-and-safety