Full Requirements Documentation

for

MNPUL8R, Release 1.0

Version 1.0 approved

Prepared by:

Keana Mowery

Vincent Orlowski

Fernando Flores

Yeong-U Lee

Alexandria Heston

November 9th, 2016

i. Table of Contents

ii. Revision History

Name	Date	Reason For Changes	Version
Patrick Lee	10/29/16	Initial draft outline creation.	0.5 draft 1
Keana Mowery	11/01/16	Insertion of information.	0.7 draft 2
Vincent Orlowski	11/01/16	Insertion of information.	0.7 draft 2
Alexandria Heston	11/01/16	Insertion of information.	0.7 draft 2
Fernando Flores	11/01/16	Insertion of information.	0.7 draft 2
Keana Mowery	11/04/16	Editing information.	0.8 draft 3
Fernando Flores	11/05/16	Editing & insertion of information.	0.8 draft 3
Patrick Lee	11/06/16	Editing of information.	0.8 draft 3
Vincent Orlowski	11/07/16	Editing of information.	0.8 draft 3
Keana Mowery	11/07/16	Insertion of information.	0.9 draft 4
Fernando Flores	11/07/16	Editing of information.	0.9 draft 4
Alexandria Heston	11/07/16	Editing of information.	0.9 draft 4
Keana Mowery	11/08/16	Final edits of information.	0.9 draft 4
Vincent Orlowski	11/09/16	Add/Edit description details.	0.9 draft 4
Patrick Lee	11/09/16	Finalizing information & document.	1.0 final draft
Keana Mowery	11/09/16	Appendix integration	1.0 final draft

1. Introduction

1.1 Purpose

This document describes the functional and nonfunctional requirements for release 1.0 of the MNPUL8R virtual reality walking and environment simulator. This document is intended to be used by the members of the project team that will implement and verify the correct functioning of the game. Unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

1.2 Project Scope and Product Features

Our virtual reality puzzle game will allow a single user wearing either an HTC Vive, Oculus Rift, and/or Playstation VR system to interact with the virtual world and solve interactive puzzles while being able to manipulate different audio music tracks via gestures through an external hardware device named, “Leap” with its “Orion” API. A detailed project description is available in section 2.

1.3 References

1. Unreal Engine Documentation: https://docs.unrealengine.com/latest/INT/
2. Leap Motion Orion Documentation: https://developer.leapmotion.com/documentation/index.html?proglang=current

2. Overall Description

2.1 Product Perspective

The MNPUL8R Virtual Reality Simulator is a game that utilizes LEAP ORACLE hand tracking hardware and SDK to allow users to interact with an immersive virtual environment in a native way. This is done by controlling the simulation via hand gestures. Using these gestures, the player can learn and explore their environment to solve puzzles and unlock new levels. The 1.0 release of MNPUL8R will be expected to have VR integration, environment controls, LEAP SDK controls, interactive objects, and puzzles for the player to uncover and solve. In following releases, more depth and functionality will be added to the game. This means adding more objects and graphics as well as dynamic functionality to some objects, improvements to efficiency, and more streamlined gameplay. Ultimately, the game's structure will be as an immersive interactive simulation and will represent this structure when complete.

2.2 User Classes and Characteristics

Player	The player is the user who is playing the game and therefore the player object is most crucial to gameplay. It will dictate camera operations, track game interactions, provide and use input from the user to interact with the game and objects within the game at large. The properties of the player object will dictate how and when a player can interact with the environment, as well as the overall experience of playing through the interactive simulated world. The Player will need to be appropriately configured to allow seamless interaction with the game environment around it. It will also need to be made with consideration of the other game objects to ensure the smoothest, most immersive player interactions. A player must be able to navigate the environment without getting stuck or breaking any functionality when playing within reasonable use conditions.
Interactive	An interactive actor is an object or sound in the game that contains properties that are able to change via interactions and input from the user. An actor can be created, destroyed, or manipulated through gameplay code via C++ or blueprint nodes. Such an interaction may include toggles, zone triggers, or timed events.

Environment	The environment objects will be those objects who's properties may be dynamically changed by user control and interaction. These will be crucial to gameplay, as some manipulations of these objects will allow players to complete a puzzle or to advance levels. Examples of the variable properties changed in these objects would be sounds, size, color, shape, positioning, and/or collision field.
Static	A static actor is an object that can be placed into the VR game that has no properties to be manipulated. Such actors may include trees, pools or water, walls, rocks, etc. These actors are used to prevent the user from "falling of" the bounds of the levels. Mostly objects that are purely for graphical purposes

2.3 Operating Environment

OE-1: The MNPUL8R virtual reality game will operate with the following VR gears: Oculus Rift, HTC Vive, and Playstation VR.
OE-2 : The MNPUL8R virtual reality game will operate with Leap Motion hand gesture tracking device attached to VR gear.

2.4 Design and Implementation Constraints

CO-1: The system's design, code, and maintenance documentation shall conform to the *Process Impact Intranet Development Standard, Version 1.3* [2].
CO-2: The system shall use the current corporate standard Oracle database engine.
CO-3: All HTML code shall conform to the HTML 4.0 standard.
CO-4: All scripts shall be written in Perl.

CO-1: The MNPUL8R game's design, code, and documentation can be found on ??
CO-2: The MNPUL8R game is developed with Unreal Engine and designed with Mudbox, Blender, Maya, and Illustrator.

2.5 User Documentation

UD-1.1: The game will have a simple form of tutorial to help familiarize the player with the environment and controls before playing

UD-1.2: The game will provide some descriptive help to the player when less intuitive controls or interactions are to be utilized at the time

UD- 2: There will be general game information beneficial to the player included in a help menu where controls and objects can be referenced to provide the player ease of use when certain game concepts are missed, misunderstood, or forgotten.

2.6 Assumptions and Dependencies

AS-1 : The user must have the required hardwares in order to play the game. Details provided in 5.1

AS-2: The game will depend upon the Unreal 4 Game Engine

3. System Features

3.1 Using the Menu

3.1.1 Description and Priority

A user puts on the hardware and opens the game, the user is confronted with a menu to create authorized identification. The user is asked to complete a series of settings and identification values so that they may play the game in a more personalized setting. Priority = High

3.1.2 Stimulus/Response Sequences

Stimulus: User performs gestureID ‘menu’.
Response: Menu appears.

Stimulus: User performs gestureID ‘down’.
Response: Menu item selection moves down one unit.

Stimulus: User performs gestureID ‘up’.
Response: Menu item selection moves up one unit.

Stimulus: User performs gestureID ‘back’.
Response: Menu moves back one phase unless on main menu.

Stimulus: User performs gestureID ‘select’.
Response: Currently highlighted selection activated.

Stimulus: User performs gestureID ‘left’ on slider element.
Response: Currently highlighted slider node moves left a notch.

Stimulus: User performs gestureID ‘right’ on slider element.
Response: Currently highlighted slider node moves right a notch.

3.1.3 Functional Requirements

gestureID.menu	The menu will appear.
gestureID.down	The highlighted selection will move down one unit.
gestureID.up	The highlighted selection will move up one unit.
gestureID.back	The highlighted selection will move back one unit, or deselect if it is currently activated.
gestureID.select	The highlighted selection will lock in as activated.
gestureID.left	The slider bar node will move left one notch.
gestureID.right	The slider bar node will move right one notch.

3.2 Playing the Game

3.2.1 Description and Priority

A user with authorized identification is placed in the first level and enters in the environment as a player. A player is tasked with completing the game through changing the environment and music. They do so by manipulating the models in our environment through hand motions registered by the hardware.
Priority = High

3.2.2 Stimulus/Response Sequences

Stimulus: User performs gestureID ‘bash’ within object hitbox.
Response: Item is activated, game initiates timer.

Stimulus: User performs gestureID ‘morph’ with crosshair within hitbox.
Response: Item is morphed, game initiates matching requirement for all objects.

Stimulus: User enters portal in maze level.
Response: User character is moved to other end of portal.

3.2.3 Functional Requirements

Player.Move:
The player will move as dictated by the input gathered from system hardware

Player.Teleport:
The game will include portal objects that, upon proximity/contact, will teleport the player object to a corresponding location.

Player.Touch:
The game will have objects that react to contact with the player object.

Player.Grab:
The game will allow the player to loot or pick-up certain game objects. This will be necessary to complete some puzzles

Player.Mnpul8:
The game will have a control scheme for environment variable manipulation. Certain environment objects will react dynamically to this control input.

Player.Do:
The game will have a simple layout for basic player-environment interactions. This is where simple I/O interactions occur.

Info.Pop
The game will provide popup instances of text/visuals to guide the user.

Game.Load
The game will load a new level when certain player-environment conditions have been met.

3.3 Create, Save and Quit Game

3.3.1 Description and Priority

A user will be able to create saved game states to track their progress, which they can save at any time. These interactions will be handled through the player menu system. Through this same menu, the player may also quit, or leave their game back to the menu, or to close the program. This way, multiple users can play independently without overwriting each other's progress.
Priority: High

3.3.2 Stimulus/Response Sequences

Stimulus: User selects "Create Game" from the main menu.
Response: User is prompted to create a new save state. Upon creation, a new save file will be made that stores the user's progress within their instance of the game.

Stimulus: User selects "Save" from the main menu.
Response: The game's save state is overwritten to the game's current conditions.

Stimulus: User selects "Quit" from the menu.
Response: Depending on whether the player is in-game or not, the game will either exit to the Title Screen, or close the game program.

3.3.3 Functional Requirements

Game.Create:
Creates an instantiation of the game for the player that is saved as a name gotten by the user during the creation process (See Game.Create.Name)

Game.Create.Name:
Process by which the user will get the name of their save state which will be then created and used for loading.

Game.Save:
Overwrites a save file from Game.Create to store and update the current game state of the player.

Game.Exit
If game is in-play, will return to Title Screen
Otherwise this will close the program

4. External Interface Requirements

4.1 User Interfaces

UI-1: MNPUL8R will provide a menu upon startup. This menu will allow the user to select from four options: *Start New, Select Level, Controls, Quit Game.*
UI-2: The system will provide the *Controls* menu page so the user may review controls for playing the game.
UI-3: The system will provide the *Select Level* menu page to skip ahead to a desired level.
UI-4: The system will provide a helpful hints when a user encounters a new puzzle task where the user must be walkthrough.
UI-5: Throughout gameplay the system will provide the user with a HUD(Heads Up Display) for displaying properties of relevant actor values.

4.2 Hardware Interfaces

HI-1: The game will use a compatible SteamVR device to provide graphical feedback for the user. This hardware typically includes some form of VR headset which can provide x, y, and z values for in-environment camera placement.

HI-2: The addition of the LEAP ORACLE hardware sensor with its appropriate SDK will get input for player hand gestures. The hardware will read the relative positioning of the player's hands and provide input feedback to be used by the game to dictate the interactions of the player object within the game environment

HI-3: A computer with appropriate computing power to handle the large resource consumption necessary to support a VR gaming environment.

4.3 Communications Interfaces

CI-1: The game system will communicate with the user via pop up windows.
CI-1.1: The system will display a pop up window when the user approaches a new object.
CI-1.2: The system will display a pop up window when the user encounters a new gesture.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

PE-1: The game shall accommodate 1 user for gameplay.

PE-2: The PC running MNPUL8R will require the following minimum requirements in order to run effectively:

- Graphics processor: Nvidia GeForce GTX970, or AMD Radeon R9 290 equivalent or greater
- CPU: Intel i5-4590 or AMD FX 8350 equivalent or greater
- RAM: At least 4GB
- Video Output: HDMI 1.4 or DisplayPort 1.2 or newer
- USB Port: One USB 2.0 or greater
- Operating System: Windows 7 SP1 or newer

PE-3: LEAP Motion controller will require two new system minimum requirements in addition to the above:

- An additional USB 3.0 port
- An increase in RAM requirement to 8GB

PE-4: The system shall display confirmation messages to users within 4 seconds after the user submits information to the system.

5.2 Safety Requirements

SR-1: The user can be exposed to the following side effects ranging from seizures, nausea and dizziness. The user should take proper precautions to prevent such side effects.
SR-2: The user can be exposed to virtual crimes or violating cyber law without acknowledging. Clear guidelines should be provided to permit or ban specific actions.

5.3 Security Requirements

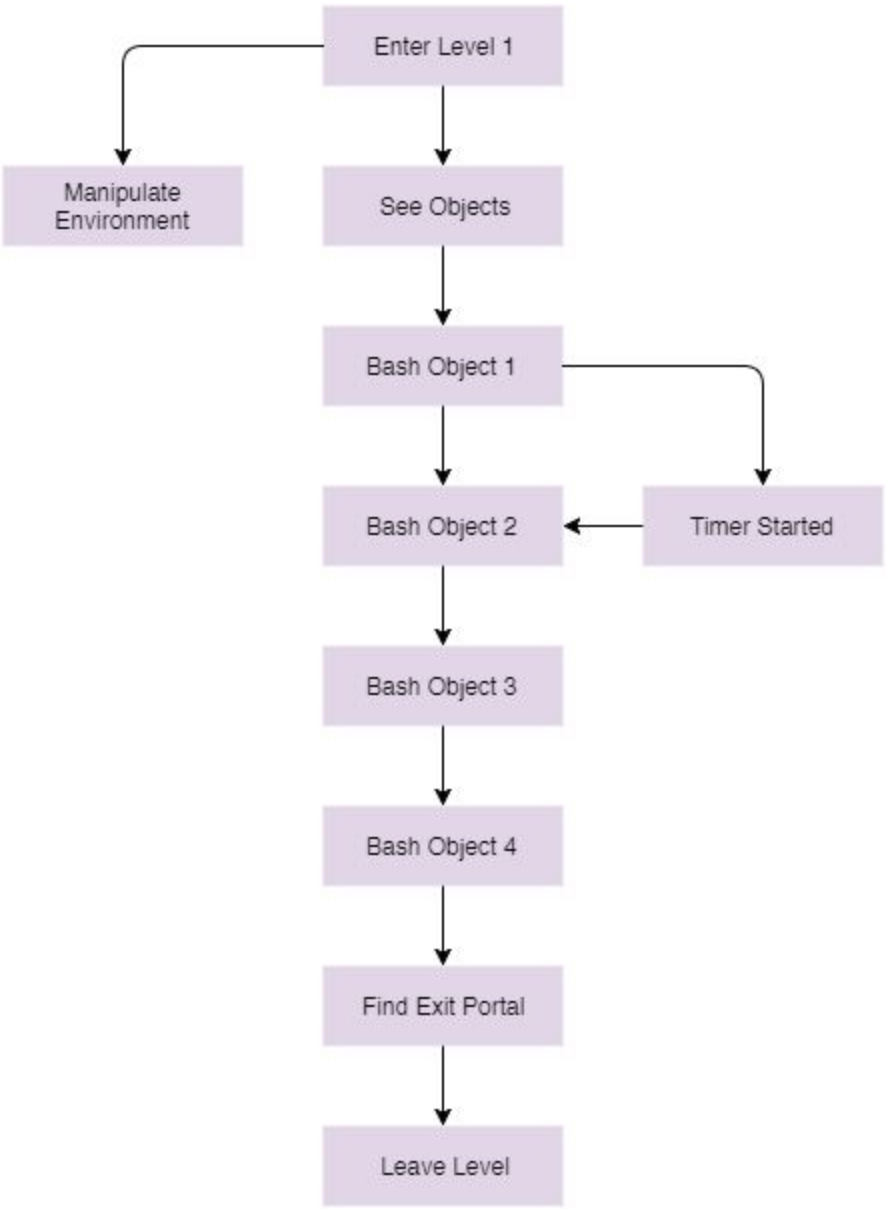
SE-1: The game source code and Unreal Project will be available to any who wish to download mod various in game actors, models, world properties, and meshes.
SE-2: The game does not provide any features that require an internet connection other than to those who wish to download the source project file and code.

5.4 Software Quality Attributes

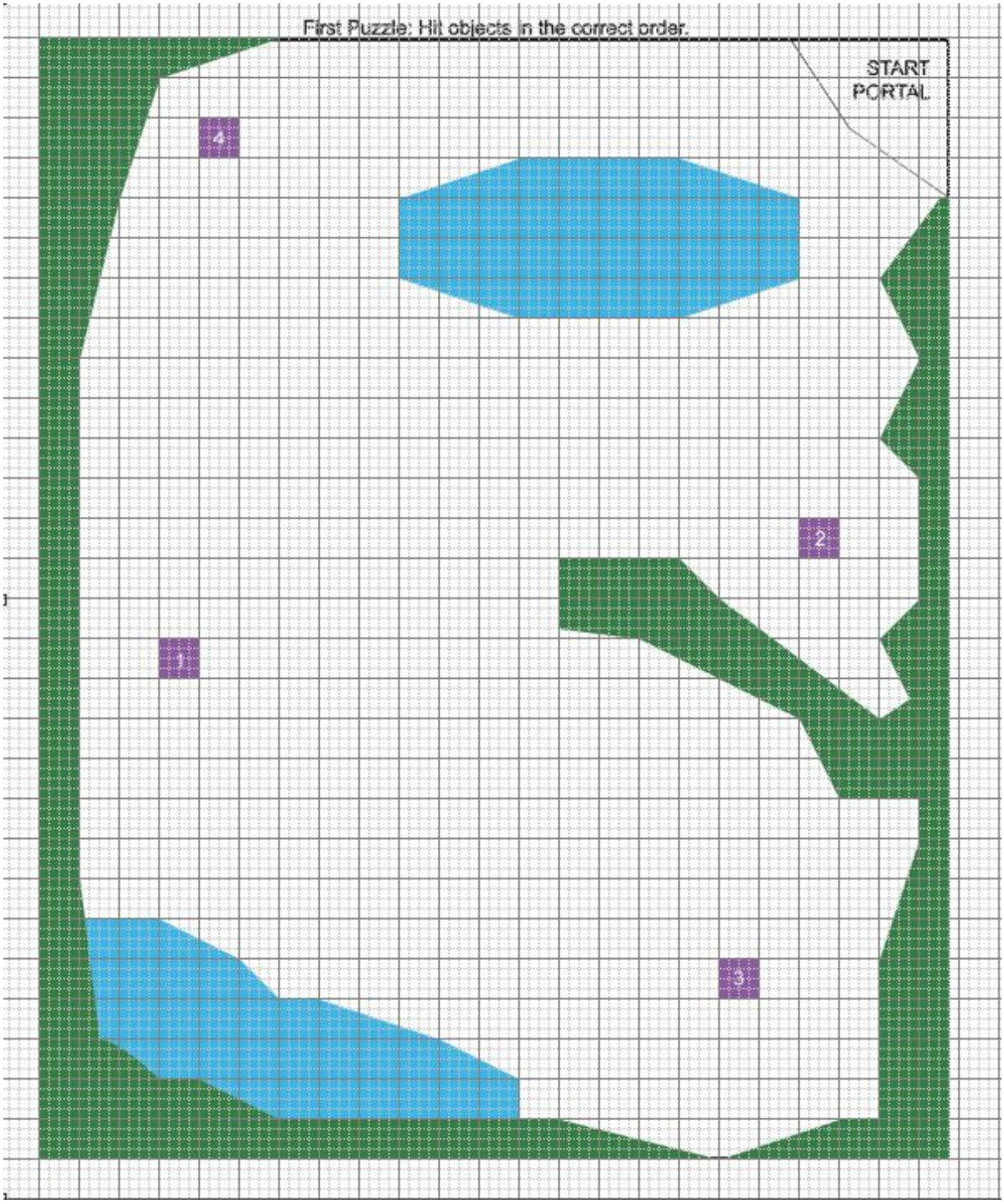
Availability-1: The game is available to any user who has the game file and right hardware.
Robustness-1: Our game doesn't require an internet connection to play. However, If the user wishes to download the source project file and code, the internet connection is required.

Appendix A

1. Gameplay Flow : Level 1



2. Game Map: Level 1



Appendix B: Level Plan

Level 1 (Blue / Green / Yellow)

Environment map bordered by trees. Appears to be a clearing in a forest split somewhat into two sections. Terrain mostly flat with small margin of variation.

User spawned at corner of map in front of cave portal. User enters the environment and see a pond and an item off to the right. As the user explores the level they will discover more items. The user must activate these items in the right order within a user-tested reasonable time limit to reveal the cave and progress to the next level.

> Objects: These are purple floating crystals, about chest level to the player, that the player bashes to activate.
>
> | | |
> |---|---|
> | Fish: | Green → Yellow |
> | Grass/Foliage: | Blue → Green |
> | Water: | Yellow → Blue |
> | Sky: | Green → Yellow |
> | Tree trunks: | Green → Yellow |
> | Forest Portal: | Blue and Purple when activated
(purple visually relates the crystals to the portal) |

Models for Level 1:

- Forest -
    - Look: Forest clearing. Light shining in from above - holes in the treetops.
    - User interaction: Flex sensors activate color shifts.
- Ponds -
    - Look: Dip in the environment that allows user to walk into
    - User interaction: user may access the ponds by walking underneath or above them (texture map for water on the bottom?)
    - Fish - Change colors
    - Tall Grass/Flowers -
        - Look: grass/flowers sticking up by the edge of the pond
        - User interaction: Flex sensors shift color scheme.
- Forest Portal -
    - Look: Dark before activated, swirling when activated.
    - User interaction: This is a teleport from Level 1 to Level 2. The user may not enter until bashing the four objects in the right order within the time limit, completing the functionality element of the level, which activates the portal for entry.

Level 2 (Purple / Red / Pink)

Environment map contained within cave for first half, outside the cave the map is bordered by trees. Terrain mostly flat with stalagmites and stalactites near walls of cave.

User spawned right in front of portal inside cave entrance. Entrance portal is black and nonfunctional. User cannot go back to Level 1.

User enters the environment and an item is immediately in view. As the user explores the level and exits the cave they will see more items. The user must use the flex sensors to alter the properties of the item to make them all match. When the items all match, the user can progress to the next level.

Ground: Red → Magenta
Cave Walls: Magenta → Purple
Sky/Cave Ceiling: Purple → Pink
Tree Trunks: Pink → Red
Grass/Foliage: Red → Magenta
Objects: Bright Blue
Tunnel Portal: Purple and Blue when activated

- Forest
    - Look: Woodsy terrain (mostly flat) with trees and leaves
    - User interaction: unsure
- Cave
    - Look: Terrain mostly flat with stalagmites and stalactites near walls of cave.
    - Functionality: Flex sensors activate color shifts.
- Waterfall
    - Look: Bright and shiny and bubbly
    - Functionality: User may walk by waterfall and interact with the colors
- Tunnel
    - Look: Darkness, Purple and Blue when activated. Once users complete the amount of interactions necessary for this level, users may be able to go through the waterfall into the tunnel which leads to Level 3

Level 3 (Purple / Yellow / Teal)

Maze Generator: http://www.mazegenerator.net

Enter the environment (tunnel) and explore the maze. The user will come across portals. The user must navigate the maze and its sets of portals correctly. Once the user passes through a set of maze portals, they may enter them again to go backward in the maze. After a few portal jumps, the user will arrive at the end of the level.

Purple walls, Teal floor and ceilings, Yellow portals?

Cave Walls: Purple → Teal
Stalactices / Stalagmites: Teal → Yellow
Ground: Yellow → Pink
Cave Ceiling: Yellow → Purple
Portal Set 1: Dark / Teal
Portal Set 2: Dark / Pink

Models for Level 3:

- Portals-
    - Look: Swirling pairs of color. Portals have different color pairs to denote matches.
    - User interaction: User teleports between sections of the maze using these portals. Final portal spits the user out at the final section of the maze.
- Maze -
    - Look: Dark.
    - User interaction: This is the main feature of this level.
- Final Room -
    - Small, dark hole in cave leads to room with technicolor crystal in center, surrounded by all objects from the game on pedestals, denoting end of game and final ascension into enlightenment.

Appendix C: Concept Art

Environment

Our environment will be created using low-poly forms in a VR environment, similar in texture and color to this mockup. This is an example of a portal inside of a cave that leads the user between one level of the game to the other. The user will reach this area of the first level and if they have completed all the necessary interactions, the portal will become available to them so as to progress in the game. This is the first true confirmation of progress, as all other user interactions will merely be an added feature of the level.

Interactive Objects

This is a photoshop mockup of an object in our environment that the user has to bash in order to progress in the first level. This model is not a part of the environment, and will be obviously interactive. It will spin and glow to denote its importance. The appearance of interactive objects is different in the two levels that

implement them.