VeaRth Games

User Manual: VR Circus

By

Brendan Baalke
Sangwoo Baek
Carson Reykdal
Isaac Shell

Table of contents

Introduction & Requirements

This manual will depict how to use and install the VR Circus game, as well as provide a list of known bugs and git instructions for retrieving the project for future developers. For a quick overview, the VR Circus is a virtual reality android game designed to work with Google Daydream-compatible devices, android version 7.1 or newer. This game does not require the user to move around the room, only arm and minimal head movement is required. Necessary materials to install and play are listed below:

- The VRCircus.apk installer file
- A computer for transfering the APK to the Android device
- A USB cable to facilitate the transfer from the computer to the Android device
- The Daydream-ready Android device
- The Daydream controller
- The Daydream headset

Git Instructions

For anyone who wishes to extend the current VR Carnival project, the Unity project is currently located on GitHub at:

https://github.com/quazemo/CSCI491-VRCircus.git (https)
or
git@github.com:quazemo/CSCI491-VRCircus.git (ssh)

If you are trying to retrieve a clone of this repository in windows, it is suggested that you install git bash for the ability to use the normal console git commands. You can find the download here: https://git-scm.com/download/win. Once installed, open up the git bash console, and navigate to the directory where you want to put the project. Enter the following command to retrieve the repository:

git clone https://github.com/quazemo/CSCI491-VRCircus.git <directory>

This should give you a local copy of the repository in the directory you specify. The directory is optional, and not specifying anything will dump the project in the current directory. From here, you will want to switch to the “Final” branch, since master doesn’t contain the complete project. Enter the following to do so:

git checkout Final

You should now have the project files, and can open the project in unity. Be warned, git doesn’t always play nice with unity files. If you decide to make a new repository, only use git for commiting work for emergency rollback purposes. Never try to merge. In all likelihood, even if you have a proper gitignore file with large file system support, it can still cause the resulting

merge to be missing files or corrupt some aspect of the project. Such being the case, be prepared for manual merging if you choose git for your repository needs.

As a final note, there is one component that is still separate from the main game. The paint toss game was causing issues when trying to integrate, and therefore it will live as a unitypackage called “PaintTossPackage.unitypackage” inside the project folder. The reason for the difficulties of integrating it are unknown, but may have something to do with our Unity version migration from 5.6.0f3 to 2018.1.3f1, as well as the extremely varied project asset structure from scene to scene. Most of the errors given when trying to integrate had to do with double references or missing definitions for the GoogleVR SDK.

Installation

Since the VR Circus game is not currently on the app store, the manual installation of an APK file will be necessary to install this program. To start this process, you will need the VRCircus.apk file and a Google Daydream-ready device. Follow the list of steps below to accomplish this:

1. On your Android device, go to the settings menu, and enter the security section (settings>security). Allow “Unknown Sources.”
2. Connect your Android device to the computer using the device’s USB cable. A notification should arrive in the device’s drop down menu. Swipe down from the top of the screen to view the list of notifications, and ensure that the USB option notification is set to “File Transfer”.
3. On the computer, navigate to your device folder, and drag the APK file into a folder of your choice.
4. On the Android device, navigate to the folder, and tap on the APK, and hit install.

At this point, the game should install, however, there is one more step before the game can be played. Enter the Daydream app, and follow the prompts to set up an account. The setup process will also provide basic safety information and tips about using the Daydream VR functionality. Once the account has been set up, exit the daydream app. You can now start the game.

User instructions

Upon startup, the daydream controller will need to be paired with the android device. Follow the on-screen prompt to do so. Once in the game, the main menu will load, and play can begin. There are currently three games in the scene. Each subsection below will describe the controls and functionality of each of these sections, as well as the main menu itself.

1. *Main Menu/Carnival Entrance:*

When starting the game, the player will be standing in front of the three desks. There is one panel on top of each desk, and when the player points the controller towards one, an image view of an individual game will display on the panel. If the user clicks the circle pad while pointing at a panel, it will transition the player to the displayed game scene.

Each panel represents the one of three mini games:

- Left Panel: Shooting Gallery scene. The goal is to knock out many targets as you can and get a high score.
- Middle Panel: Catapult Frenzy. Build your own base and knock out the enemy's base using a catapult.
- Right Panel: Paint Toss. Use a paintball to knock down generated targets.

2. *Paint Toss:*

When entering the paint toss game, the player will see a game board placed in front of their view. The player is given a paintball that they can then throw at some visual menu boards to select difficulty and game type. These menu board options are currently in place on top of the gameboard. The paintball is spawned directly in the users hand when they click the generic select button on the controller. The position of the paintball is based upon the 3D world-point of the controller with reference to the users headgear location.

After selecting gameplay options, the player is presented with a challenging, randomly generated game board with targets (under development). A timer is placed near the edge on top of the screen indicating the player's remaining time to complete the challenge. A scoreboard shows how well the player is doing. This timer and end game condition is still in development.

Optional Addition: There is a third party open source file called DecalPaint, which is a script for adding splatter paint effects to destination objects. This allows the paintballs to actually get destroyed and create a splattering effect on the object that they collide with.

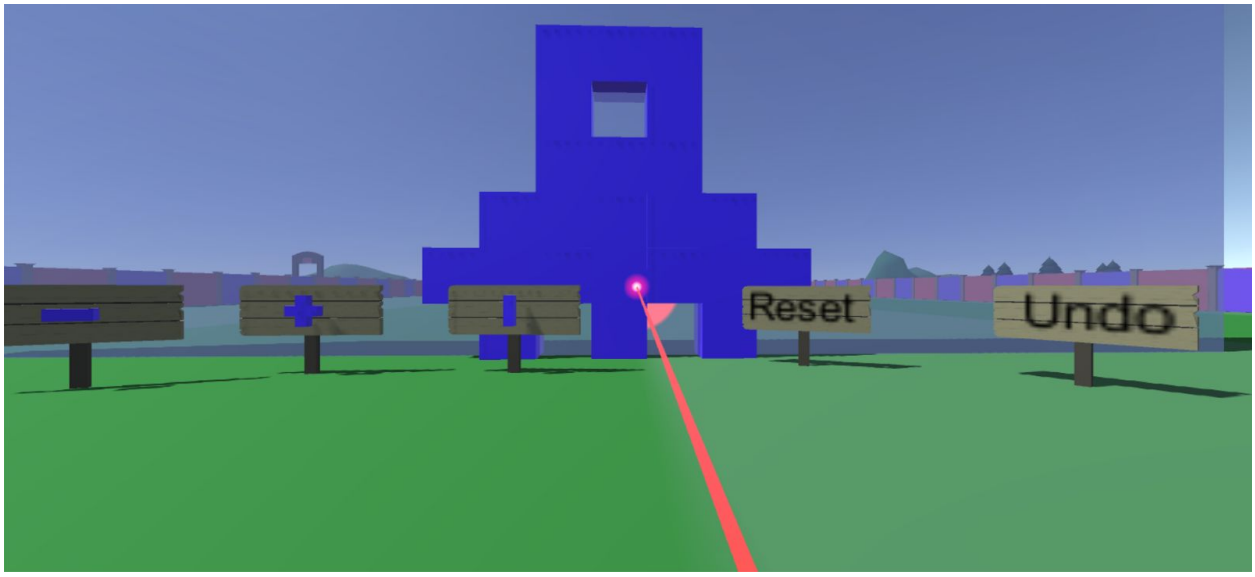
3. *Catapult Frenzy:*

Upon loading in to the game room, the player will be in a side area with a laser pointer and several signposts. The laser pointer can be aimed by changing the angle of the controller. When you press down on the touch button, the red light of the laser pointer will turn green to indicate a click.

There should be four signposts to the left of the player. Each of these correspond to a different difficulty and when you hover over them with the laser pointer and press the touchpad, the sign's text will change color and the difficulty will be changed. On easy difficulty, the AI has a large degree variance in its aim and will likely miss most of the time. On medium, the AI becomes slightly more accurate but still misses more than it hits. The Hard AI will hit roughly every other time, and even more so if your tower is tightly clustered. "Plz No" difficulty will almost always hit.

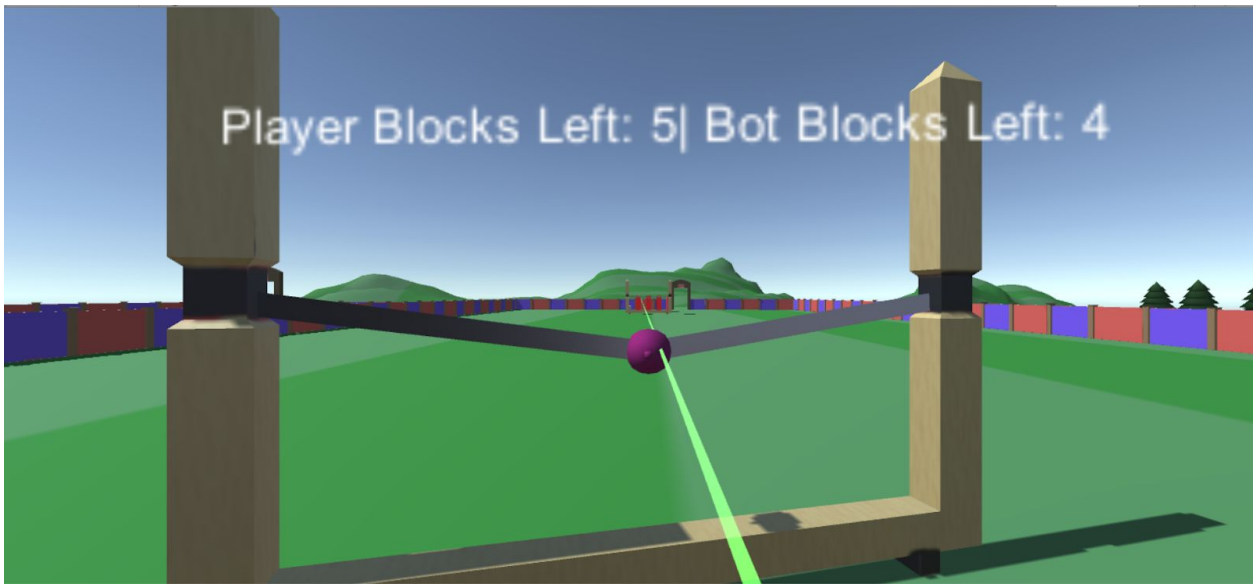
To the right, there will be a start sign which starts the game and an exit sign which returns the player to the lobby area. Upon pressing start, the player will be teleported inside the main game area and will be faced with a new set of signs and a large transparent rectangle. To the left, there will be signs, each with a picture of a block on them. Clicking and dragging these blocks onto the rectangle will align them to a grid. Releasing the touch button will place them (Note that a straight horizontal block cannot be placed on the bottom 2 levels). Currently you can only place 5 blocks (that can easily be changed in PlayerController). Figure 1 shows this tower builder setup.

Figure 1: Catapult Frenzy Build Phase



To the right, there are a few more signs. Hitting undo will refund you a block and remove the last one you've placed. Hitting Reset will send you back to the lobby area. If you've placed at least one block, hitting Start will move the user forward and begin the catapult phase.

Figure 2: Catapult Mode



You will notice the AI has also laid out a set of blocks on the opposite side of the map, as well as the appearance of your catapult with a purple water balloon in the center. Pressing the touchpad while hovering over the balloon will commence your shot. Moving your thumb around the touchpad will aim the catapult with forward and backward indicating power and left and right being angle. Releasing the touch button will releases the water balloon, and after it hits a target or falls out of the world, the AI will take its turn. This will continue until one player has no more standing blocks, at which point the text in the center will declare one player the winner. There will be a restart button to the left that allows the player to start over from the beginning.

4. *Shooting Gallery:*

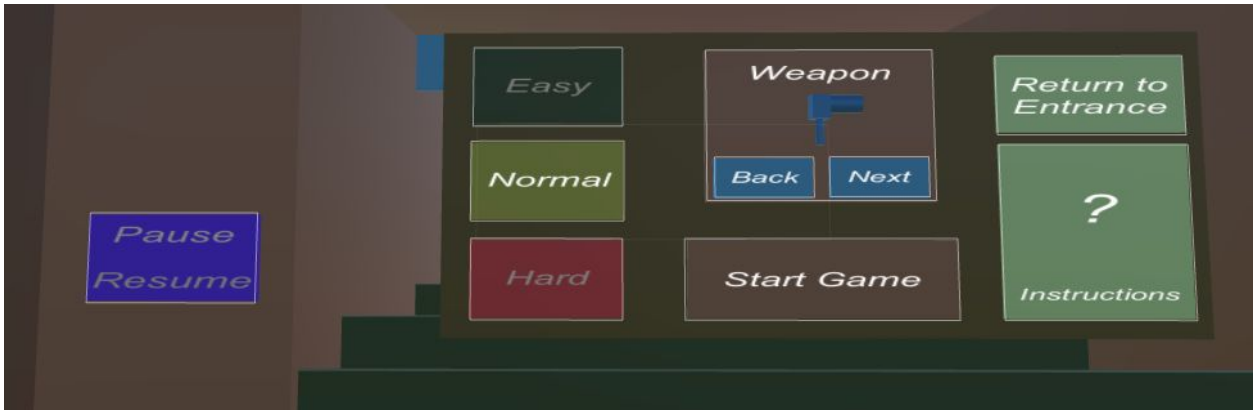
Upon entry into this game, the player will be holding an egg shooter. To fire this weapon, simply press down the circle pad. Aiming the weapon is handled by the motion of the controller. Should the controller or the camera view seem misaligned, point the controller forward and hold down the home button.

In front of the player should be a shooting gallery building and a menu sign. Take aim at the buttons on this sign, and shoot them to activate menu actions. Below is a list of what each of the buttons do when shot. Figure 3 shows the menu view.

- Easy Button: This sets the game mode to easy. In easy mode, target spawning will not speed up very quickly, and there will be less targets to shoot in total.
- Normal Button: This sets the game mode to normal. Targets in this mode will spawn faster as the game progresses. Significantly increases the number of targets total in comparison to easy mode.

- Hard Button: This sets the game mode to hard. The targets in this mode spawn faster as the game progresses, and increases the total number of targets yet again. The spawn speed increase interval is significantly shorter than normal mode.
- Weapon Back/Next Buttons: These buttons change the color of the egg shooter from a select pool options.
- Start Game Button: This button starts the game. The menu will collapse, revealing the shooting gallery spawn rows behind. Targets will start spawning a few seconds after the menu collapses.
- Return to Entrance Button: This button will cause the game to transition back to the main menu.
- Instructions Button: This button is currently unused. In a future update, this button will give voiced instructions on how to play the game, which would also include spawning a controlled tutorial gun to show younger players which buttons are which.
- Pause/Resume Button: This button will stop the game and call the menu back into view, or cause the menu to vanish and resume a game in progress. A game can be restarted by pausing the game and hitting the start button.

Figure 3: Shooting Gallery Menu



When the game is started, targets will appear at varying distances along the spawn rowes. Shooting these signs will accumulate points which are displayed below the shooting gallery window. The game ends when there are no more targets left to shoot, and the target counter in the back is 0. Figure 4 shows both the score and target counter displays.

Figure 4: Shooting Gallery Game View

Known Bugs

Below is a comprehensive list of known bugs in the VR Circus software. If applicable, the bug descriptions will also include a way to reset or otherwise deal with the bug when it arises.

- Shooting Gallery: Toggling the game to the paused state close to the end of the hide animation for the menu causes the game to pause with no menu. This can be circumvented by resuming the game and pausing again.
- Paint Toss: The tossing mechanic for the game is not broken, but it is a little buggy when it comes to how smooth the physic reactions affect the paintball toss. With further examination this could be fixed.
- Catapult Frenzy: The score text doesn't reset when the game is restarted.

Development Challenges

Below are the primary challenges that were encountered in the project thus far. If possible, solutions and workarounds will be provided. This list is not a comprehensive list of all problems, however. It's only composed of issues that future teams may find useful or enlightening.

1. Git - As mentioned above, there was a lot of trouble trying to use git for collaborating code. The primary functionality of being able to merge and rebase branches didn't work well if at all. As a result, working together became much harder, since a lot of our schedules didn't line up from quarter to quarter. To combat this, we separated our responsibilities by game/scene so that there were few dependencies going forward. Some continued to use git simply for saving their work, but the team was essentially free

to use any repository of their choosing for this purpose. If another attempt were to be made at using git for more than just commits, perhaps limiting scope to just tracking scripts and primary prefabs would be the way to go.

2. Obtaining testing hardware - The google daydream set obtained from the student technology center at WWU was very hard to get a hold of. Originally, it was offered for rental 3 days at a time since it was a new, unlisted item with no scannable barcode. However, after one member checked it out and returned it for the apparent first time ever, they released it on a 2 week rental period. From the looks of the shelf it was stored on, they only have 2 to give out, and both were checked out when we were ready to start implementing VR a couple days later. Bottom line, check out the devices early, and try to make reservations. For reference, we tried to get a device at around the half point in the quarter, and we got a device on Wednesday before dead week. This left us with barely enough time to implement and test VR.
3. Instantiate Problems - In Unity, there is a function called Instantiate that spawns an instance of a prefab object into the current scene. For those new to unity, it can cause some weird behavior if you are unfamiliar with the exact execution order between it and the instantiated object's start() and awake() functions. If you need to use Instantiate, be sure to brush up on the execution order to ensure script initialization goes smoothly. For an example of what can happen, Instantiate has an optional argument to automatically set a parent object to the instantiated one. The instantiated object had an awake function that expected to initialize an object based on the parent. Awake is called directly after instantiate according to the documentation, and is supposed to be used to set up references between scripts so that start may use those references. Unfortunately, instantiate doesn't set the parent before awake, so the needed parent reference was null. This was fixed by instantiating a prefab without the script, then attaching one later.
4. Asset Bookkeeping: During the final integration, asset folders got out of hand really fast. There was no naming scheme that we afore agreed upon, and as a result, some script folders merged with others while some created separate ones. Very annoying conflicts can also arise if some random imported class defines something already within the current project. Unity doesn't always tell you where both references are all the time, so if your hierarchy is a mess, the problem compounds, and you will end up searching the whole asset tree. Keep your asset hierarchy orderly, and backup before every import just in case. Come up with a plan for what folders specific assets will be stored in, such as project scripts, materials, and pefabs.