University of Szeged

Faculty of Science and Informatics

Implementation of a Virtual Reality-based garden simulation

Bachelor's thesis

Author:

Bársony Daniella

BSc in

Computer Science

Supervisor:

Dr. Varga László Gábor

senior lecturer

Szeged

2019

Contents

Task proposal

The task was to implement a virtual environment where the user can tend to a garden in his own home. Part of the task was to choose a VR device (Google cardboard, HTC Vive, Oculus Rift, etc.) which is capable of visualizing a virtual world. With the chosen device then implement the game itself, which has the features stated below:

- The locale of the game is a garden, which the player can move around in.
- There are plants which reside in the garden.
- Plants in the garden will follow the rules of nature even if not exactly as in reality (For example: growing, they can dry out if the water level is not sufficient, fruits can grow on them, etc.)
- In the garden it is possible to have some interaction with the plants, and tend to them with the given tools.

Summary

• **Denomination of the topic:**
Implementation of a garden that resides in a virtual world

• **Wording of the given task:**
The task is to implement a virtual garden. Where players can tend to the plants in 3D space. This includes watering, pruning, using pesticide if the plant gets infected, using fertilizer on the ground. In the case when our plant is happy, which means that the water is sufficient, the ground is fertilized enough and that it is not infected, will grow into a full-blown plant, break out into blossom, and bear fruit.

• **Solution:**
I made a framework for the plants, that made it easier to manipulate them through. I designed a basic User Interface for the player to get and delete plants. I implemented some basic game logic for the garden itself.
In the end I used the VRTK and SteamVR plugins that are available on GitHub.

• **Devices and methods used:**
The device of my choosing was the HTC Vive. The platform that I used was Unity version 2018.2.10f1. As for the Integrated Development Environment(IDE), I chose Visual Studio Code. C# as the programming language. Furthermore I found 2 plugins on GitHub named VRTK and SteamVR.

• **Accomplishments:**
I finished the game as described. There are x different types of plants, which we can tend to. X number of plots where we can place a plant of our liking then tend to them.

• **Keywords:**
Unity, 3D, Virtual Reality, Garden

Magyar nyelvű összefoglaló (Hungarian content summary)

A Virtuális valóság egy olyan élmény, amit egy számítógép generál és egy teljesen szimulált környezetben van. Nem összekeverendő a kiterjesztett valósággal, ami a már létező valósághoz ad hozzá. Egy harmadik, a szimulált valóság, ami hasonlít a virtuális valósághoz, de a felhasználó nem tudja megkülönböztetni a szimulációt a valóságtól. Erre láthattunk egy példát az ikonikus Mátrix című filmben, ahol az emberek nem tudták, hogy az egész életük egy szimuláció.

- **VR technológia:**

Ebben a részben a VR történelméről van szó. Kezdve az 1800as évektől, amikor Robert Barker ötletével. Le szerette volna festeni Edinburgh városát egy félkör ív alakú festményen, először röhelyesnek gondolták ezt az módot. Később egy teljes kör-ívet alkotó képet festett, amit elnevezett panorámának.
Utána 1838-ban Charles Wheatstone publikált egy tanulmányt, miszerint az emberi az a jobb és a bal szem által látott 2-Dimenziós képből alkot egy 3-Dimenziósat és így érzékeli a teret. Ezzel a technikával dolgoznak a VR szemüvegek is mostanában, külön képet vetítenek a jobb és a bal szemnek, amik egy picit el vannak tolva, így térhatást érnek el.
Sok kísérlet volt még ezután többek között egy repülés szimulátor, amivel a második világháború idején több mint 500.000 pilótát képeztek ki. Nem csak képet vetített, hanem imitálta a turbulenciát is motorok segítségével, amik mozgatták a kabint. Morton Heilig is kitalálta a Sensoramat, ami szintén nyújtott mást is a kép mellett. Ide tartozott a 3D hang, ventillátor és egy szag generátor is. Erre az eszközre több kisfilm is készült, amit ugyúgy Morton Heilig csinált.
Az első HMD(fejen viselt kijelző) ugyanúgy Morton Heilig ötlete alapján látott napvilágot. Később ez alapján Philco vállalatnál két mérnök elkészített egy olyan HMD-t, ami követi is az ember fejmozgását.
Újabb nagy név ezen a területen, Ivan Sutherland, aki letette az alapkövét a virtuális valóságnak 1965-ben, azzal, hogy megfogalmazta mi az irány elve ennek az ágazatnak.

Ebben szerepelt egy olyan kijelző, amit nézve a felhasználó nem tudja, hogy valóság vagy szimuláció van előtte. Továbbá egy olyan virtuális világ, amit a 3-Dimenziós kijelzőn keresztül látunk, valóságos hangzás és interakcióba lehet lépni az ottani világgal. Mai napig ezt a koncepciót követi ez az ágazat.
1987-ben a VR-t felkapta az ipar, a VPL(Visual Programming Lab) több eszközt is kifejlesztett, ami ehhez tartozott, például a DataGlove és egy HMD. Ők voltak az elsők akik el is adták ezeket.
1991 körül próbáltak elérni nagyobb közönséget ezzel a technológiával és arcadekben csináltak többjátékos módban játszható gépeket, amik VR-t használtak. Ezekben valós időben történtek az események és így tudtak vele játszani a játékosok.
A fűnyíró ember egy 1992-es film, ahol még több embert próbáltak elérni a virtuális valóság ötletével. A VPL által kifejlesztett eszközöket használták a filmben is és inspirációt a filmhez is a VPl történetéből merített a rendező, mint ezt bevallotta később.
A SEGA látott fantáziát ebben a technológiában és megpróbálta kifejleszteni a sajátját az akkori konzoljukhoz, és Genesisnek nevezte el. Be is mutatta 1993-ban a Consumer Electronics Show-n. Sajnos ez a projekt nem jutott túl a prototípuson, pedig 4 játékot fejlesztettek is rá.
Pár évvel később a Nintendo is kapott az alkalmon, és kifejlesztette a Virtual Boy-t. Ez az eszköz ki is került a forgalomba Japánban és Észak Amerikában 180 dollárért, így már elérhető volt az emberek számára is ez a technológia. Sajnos nem volt kényelmes így nem sokkal később a cég beszüntette a gyártását a szemüvegnek.
A nagy ugrás a Mátrix című film volt, ami mára már alap művé nőtte ki magát. Az emberek egy szimulált valóságban éltek és nem is tudtak róla.

Source: `https://www.vive.com/us/product/vive-virtual-reality-system/`

Figure 1: HTC Vive

A jelenben a HTC Vive, Oculus Rift, Playstation VR és a Google Cardboard uralja a piacot.

Az Oculus-t 2010-ben Palmer Lucky

Introduction

Chapter 1

VR technology

Virtual reality is an experience that is generated by a computer, in a simulated environment. It is not to be confused with augmented reality that adds nonexistent things to the real world, neither to be confused with simulated reality that is a dream which is far away with today's technology. It would be a reality which is 100% simulated and can not be differentiated from the real world hence fooling the person in it to thinking that it is in fact reality. We've seen it before in movies(to mention one: The matrix).
In my humble opinion virtual reality could eventually lead up to the simulated one, even if it looks impossible to reach for now. Obviously the technology for it is still light-years away but it could be a beginning. If this is a good or a bad thing that's something that only time can tell.

1.1 Beginnings

Humans played with the thought of virtual reality and how to achieve it way before we had computers. Starting from the year of 1800 until now there were countless experiments. With some hiatus every now and then but from starting at 1965 development has really sped up. At that time it was Ivan Sutherland who set the basis of virtual reality should be. That was the point when people got really interested and invested in this topic, researching it even more than before. [1]

1.1.1 Panoramic paintings

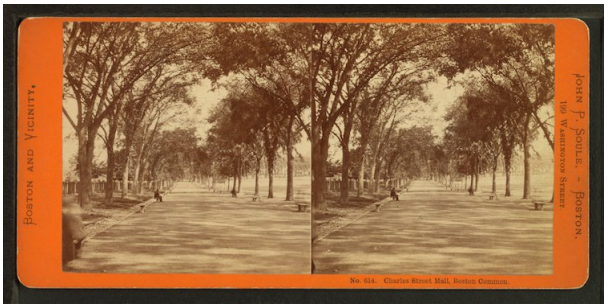
Robert Barker(1739 - 1806) who was an English painter with Irish ancestors, who came up with the idea of painting Edinburghs city on a half-circle view painting. He then shown these pictures to Sir Joshua Reynolds, who said these pictures were not practical. Robert

[1] $https://www.vrs.org.uk/virtual-reality/history.html$

Barker did not give up and later on he made a full-circle view picture too and put it up for exhibition in Archer's Hall. The name of panorama came from the Greek "pan" which means all and "horama" meaning view in 1792. [2] [3]

1.1.2 Stereoscopic pictures

In 1838 Charles Wheatstone published a study, that our brain fuses two 2-Dimension pictures into a 3-Dimensional one. That's how it perceives depth and space. Later on in 1839, Willian Gruber, then in 1849 David Brewster made glasses for the stereoscopic pictures. These glasses could be familiar already from the cheaper alternatives to today's VR, just look at Google cardboard for example which just places the pictures of the left and right eye next to each other.



Source: https://www.vrs.org.uk/virtual-reality/history.html

Figure 1.1: Stereoscopic picture

1.1.3 Flight simulator

Edward Link in 1929 invented the "Link trainer", which wasn't only using picture to stimulate senses, but used motors to simulate turbulence and other disturbances that could occur mid flight. In the beginning only theme parks ordered the device from him. In the end, United States Army Air Corps bought 6 of these and trained over half a million people with it to be an airmen during World War II.

1.1.4 Morton Heilig's Sensorama

Mid 1950 Morton Heilig, director came up with the idea of a theater which was dubbed the Sensorama. He could build a prototype of it by 1962. His idea was to not only stimulate the eyes through pictures and ears through sound but to involve all our senses so the person could get really absorbed in the experience. It came with stereo-sound system,

[2] $https://www.libraryireland.com/irishartists/robert-barker.php$
[3] $https://en.wikipedia.org/wiki/Robert_Barker_(painter)$

scent generating device, fans, vibrating chair and stereoscopic colored displays. They even made 6 short movies for the Sensorama including a Coca Cola advertisement that he shot and edited himself. He patented the Sensorama under US Patent #3,050,870.

1.1.5 Head Mounted Display/HMD

Another thing that we can thank Morton Heilig for, in 1960 he had a project called Telesphere mask. This was the first time that we've seen a Head Mounted Display(HMD). There was no interaction or movement tracking. The device had a 3-Dimensional widescreen display and stereo-sound. He patented it under Patent #2,955,156.

1.1.6 Movement tracking HMD, Headsight

In 1961 two Philco Corporation employees who were engineers developed the first movement tracking HMD and called it Headsight. It had a display for each eye and magnetic movement tracking, that was linked to a closed circuit camera. It was designed, so that the army could observe dangerous situations with its help so this was a surveillance system. The motion tracking was that if the person rotated his head left or right the camera was following the movements.

1.1.7 The concept of Virtual Reality

Ivan Sutherland described a concept in 1965, that was about a display which would represent reality so closely that a normal human being will not be able to differentiate reality and simulation. In it was virtual world, that can be perceived through a 3D display and sounds effects that made the person feel more like it's reality and tactile feedback. A computer that is generating that world and keeping it up to date in real time. Moreover the ability to interact with whatever is inside that reality in a way that does indeed feel realistic.
This was the concept that to this day people keep in mind when designing devices and so on for virtual reality.

1.1.8 Sword of Damocles

Another thing by Ivan Sutherland and his student Bob Sproull, in 1968 was the first AR/VR HMD that was linked to a computer and not a camera. This device was huge and scary looking and it was too heavy to wear it comfortably so they came up with the idea to suspend it from the ceiling. The user needed to be strapped into the device. The computer graphics were primitive wire-frame rooms and objects.

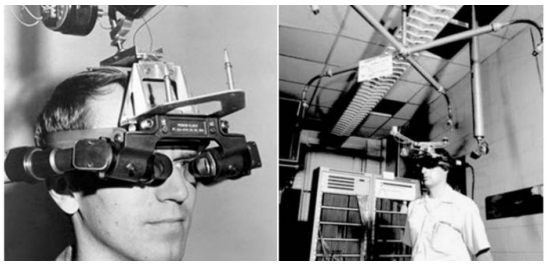


Figure 1.2: Sword of Damocles [1]

1.1.9 The VR is born

Was 1987 when the founder of VPL(Visual Programming Lab), Jaron Lanier came up with the term virtual reality. Through his company he developed many devices for this new field, like DataGlove and EyePhone HMD. They were the first company who sold VR goggles. This was a really big step for this field.

1.1.10 Virtual, multiplayer arcade machines

In 1991 VR could reach a wider audience, even if it wasn't available for home usage yet. They popped up in arcades. The players wore HMDs, that had 3D stereoscopic displays. The games themselves were running in real time, which meant that their latency was less than 50ms. There were even games where they could implement multiplayer mode, for example Dactyl Nightmare.

1.1.11 The Lawnmower Man

This movie came out in 1992 that introduced the concept of virtual reality to even more people. The movie was based on the work on Jaron Lanier who was the founder of VPL. doctor Angelo who was played by Pierce Brosnan a big name at the time, was playing with therapy that was based on virtual reality to help a mentally impaired boy. They used the devices that were made by VPL itself and the director Brett Leonard did admit that he drew inspiration for this movie from companies like VPL.

1.1.12 SEGA

SEGA announced a VR headset, named Genesis for its console in 1993, at the Consumer Electronics Show. This prototype was able to track the movements of the head, had stereo sound and LCD displays built in it. The company wanted to release it but sadly they hit a wall that they could not overcome, so it never reached customers. There were even 4

[1] Source: `https://vrroom.buzz/vr-news/guide-vr/sword-damocles-1st-head-mounted-display`

games developed just for this HMD. 2 years later even Nintendo tried to come up with something like this.

1.1.13 Nintendo Virtual Boy

Figure 1.3: Virtual Boy [2]

It was a 3D video game console, which was advertised to be the first portable game console, that was capable of rendering true 3-Dimensional graphics. First came out in Japan then North-America for 180$. Despite the efforts of Nintendo, like trying to lower its price, this project proved to be a failure. Was not comfortable and it had problems with the colors too. These were the causes that made the company stop manufacturing the said console.

1.1.14 The Matrix

It was 1999 when Matrix the movie came out, which later on grew to be a cult film. In it the characters lived in a simulated world, which looked like just our everyday life, where almost no one knew that they lived their whole life in a simulated world. Even before this film there were ones who were playing with the idea of virtual or simulated worlds, just like Tron or The Lawnmower Man, but this one was such a big hit that it introduced the idea of virtual worlds to people.

1.2 Present

This generation is dominated by HTC Vive, Oculus Rift, Project Morpheus on the high-end, while the low-end has devices as the Google cardboard and other headsets that use your phone as a display.

[2]Source: `https://www.nintendo.co.uk/Iwata-Asks/Iwata-Asks-Nintendo-3DS/Vol-1-And-That-s-How-the-Nintendo-3DS-Was-Made/2-Shigeru-Miyamoto-Talks-Virtual-Boy/2-Shigeru-Miyamoto-Talks-Virtual-Boy-229419.html`

1.2.1 Oculus Rift

In 2010 Palmer Lucky designed the first prototype of the Oculus Rift, which was capable of tracking the movements of the head and having a 90 degree field of vision. 2012 they started a Kickstarter campaign for the development of Rift. Later on Facebook purchased the whole company. The Rift right now has 2 PenTile OLED displays, 1080x1200 resolution per eye, 90Hz refresh rate and 110 degree field of view.
It's not only capable of tracking the movements of the head but position too. Moreover it has integrated speakers which provide a 3D audio effect. It has multiple accessories for example the controllers and sensors. The movement tracking sensor is named Constellation, which is not only capable of tracking the headset, but the accessories too. External infrared sensors are there to optically track compatible VR devices.
The Oculus supports third-party peripherals too, so it provides an API that people can use.

1.2.2 HTC Vive

Figure 1.4: HTC Vive [4]

By 2013 Valve join in on the virtual reality business and they had a break through with low-persistence displays, that had almost no latency and smear-free. Oculus used this technology too and used it in later developments.
HTC and Valve announced the development kit of HTC Vive in 2015.
This headset uses "base stations" for tracking which are called Lighthouse, that you can mount on walls and uses infrared light too. The consumer version came out in June 7th of 2016.
The **headset** has 90Hz refresh rate and 110 degree field of view. The device is packed with two OLED panels, one for each eye, that has 1800x1200 resolution for each of the displays just like the Oculus Rift.The headset comes with a front facing camera for safety reasons, so the user will not bump into moving or static objects in the room. It has a G-sensor in it, which tracks the acceleration and a gyroscope so it can determine the orientation, furthermore a proximity sensor too.
The **contollers** have different input methods. Dual stage trigger, track pad, grip buttons. It has a battery time of 6 hours. On the front of the controllers, across the rings there are 24 infrared sensors that the base stations use to detect the location of said devices. SteamVR tracking system is used to track the controller location to a fraction of a millimeter, with update rates of 250Hz to 1kHz.
Vive Tracker is an accessory used for motion tracking. They designed it, so the user can

[4]Source: `https://www.vive.com/us/product/vive-virtual-reality-system/`

attach it to anything and the Lighthouses will detect them and track their movements.

1.2.3 Project Morpheus/Playstation VR

Project Morpheus was the code name for this project during development. It was released in October 2016 for the Playstation 4.
The latest release of the headset comes with OLED displays, a resolution of 960x1080 per eye, the refresh rate can be switched depending on the developers to native 90Hz, native 120Hz and there is a third, where the 60Hz would be displayed as a 120Hz using a motion interpolation technique(new frames are generated between existing ones). Field of view is only 100 degrees. As for tracking it has six-axis motion sensing system (three-axis gyroscope, three-axis accelerometer). [5]

1.2.4 Google Cardboard

Last we have the Google cardboard. This is one of the low-end "devices", as the name says it is a piece of cardboard, even though there are more stylish ones. All it does is hold your phone, on which you can download an app. The result is a stereoscopic image with a wide field of view. Developers can use VR View, an expansion of the Cardboard SDK allowing developers to ember 360-degree VR content on a web page or in a mobile app on pc, android and iOS. The HTML and JavaScript code from web publishing VR content is open source and available on GitHub. [6]

[5] https://www.playstation.com/en-us/explore/playstation-vr/tech-specs
[6] https://en.wikipedia.org/wiki/Google_Cardboard

Chapter 2

Unity

A game engine is a software-development environment designed for people to build video games[1]. The core functionality being rendering 2D and 3D graphics, physics and collision detection, sound, scripting and a few more. These tools are to help the developers by simplifying the development process. There are quite a few game engines out there and they use different primary programming languages like C++, C, Lua, C# and so on. The bigger ones being Unity, Unreal Engine and CryEngine.

2.1 What is Unity

Unity is a cross-platform real-time game engine developed by Unity Technologies which was announced first in June 2005. It used to support JavaScript as its programming language but now it's deprecated and only supports C#. Supported platforms are iOS, Android, Windows, Mac, Linux, WebGL, Playstation4, Xbox One, Nintendo 3DS and much more adding up to 25+. The engine is written in IL2CPP (Intermediate Language To C++), which provides native C++ performance. Unity has XR, which is support for VR and AR and MR development. Over 91% of HoloLens experiences are Made with Unity. This game engine support both 2D and 3D game development. We have an asset store, where we can either purchase or use free premade scenes, prefabs, models and so on that anyone could need in the future for development.
Starting with 2016 Unity started to offer cloud based services for the developers, which are: Unity Ads, Unity Analytics, Unity Certification, Unity Cloud Build, Unity Everyplay, Unity In app purchase, Unity Multiplayer, Unity Performance Reporting, Unity Collaborate and Unity Hub. [2] [3]

[1] https://en.wikipedia.org/wiki/Game_engine
[2] https://unity3d.com/unity
[3] https://en.wikipedia.org/wiki/Unity_(game_engine)

2.1.1 What does it offer

Unity Editor offers multiple tools that helps the developer. Rich and extensible editor, that enables rapid editing and iteration in the development cycles for example Play mode where the developer can check the game in real-time. Makes development a lot easier with Prefabs which are preconfigured Game Objects. An intuitive and configurable User Interface. It even has its own physics engine which supports NVIDIA PhysX for highly realistic and high-performance gameplay. For collaboration it has its own version control, which helps the developers.

2.1.2 Scenes

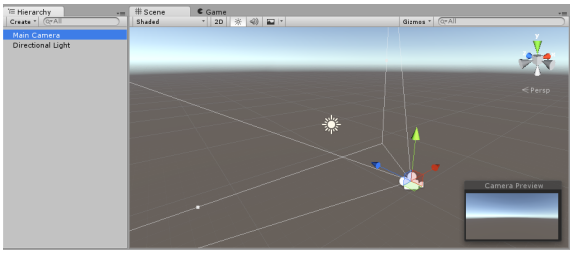


Figure 2.1: Unity Scene [4]

A scene contains the environment and the menu of your game similar to a different level. You can place your decorations, player character, User Interface and whatever you like helping you build your game in pieces. You can make multiple scenes and switch between them using the Unity API in your own scripts. Multiple scenes can be loaded too at the same time if need be.

2.1.3 Game Objects

A GameObject is the most important concept in the Unity Editor. Every object in your game is one starting from characters to lights and cameras and so. They can't do anything on their own, you need to give them properties. To give them these, you need to add components to it. You combine the different elements until they add up to the object that you are working towards. For example if you want a Light object, you need to attach a Light component to the GameObject.
GameObjects have transforms. The transform component determines the Position, Rotation and Scale of the given object in the scene. Every GameObject has it. [5] [6]

[4] https://docs.unity3d.com/Manual/CreatingScenes.html
[5] https://docs.unity3d.com/Manual/GameObjects.html
[6] https://docs.unity3d.com/Manual/class-Transform.html

2.1.4 Prefabs

This system allows you to create, configure and store a set GameObject for later use with all the settings and changes you made on it. Any edits you make on the Prefab will be reflected to the instances of that given object. The prefabs can be nested inside other Prefabs to create complex hierarchies of objects that are easy to edit.
However the instances of a given Prefab don't need to be identical you can make small edits on each of them that need to be different somehow and it won't be reflected on the parent. A good way to use a prefab could be a player character that is the same in different scenes so you don't have to make it from scratch or a weapon with projectiles and unique scripts attached to it that would be too much of a hassle to remake every time you need to use it.

2.2 VRTK - Virtual Reality Toolkit SDK

This asset can be found either on the Asset store or on GitHub. Depending on which SteamVR version the developer is using, he might need to use the GitHub master version, because the one on asset store stopped working with newer SteamVR versions.
This asset is a collection of useful scripts and prefabs to aid building VR games, for example to help you interact with GameObjects. After the developer imports the VRTK asset, it has a `VRTK_SDKManager` script that can be attached to a GameObject in the scene. It supports almost all VR devices. [7]

2.3 SteamVR

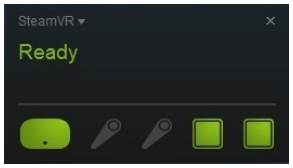


Figure 2.2: Steam VR UI[8]

This plugin is made and maintained by Valve. They want to help developers so that they can develop games with one API that supports all the popular VR headsets. It manages three main things:

1. Loading 3D models for VR controllers

[7] `https://vrtoolkit.readme.io`

2. Handling the input from those controllers

3. Estimating what your hands look like while using said controllers

SteamVR has an Interaction System example to show you how to set up a world that you can interact with. To use it the player needs to have SteamVR runtime installed, which can be download from Steam. This will be started up whenever we switch to VR mode. [9]

[9] https://valvesoftware.github.io/steamvr_unity_plugin/

Chapter 3

Implementation

As I have mentioned in my earlier chapter I chose Unity Engine to make my thesis. This chapter will describe the details of coding, getting familiar with SteamVR and VRTK Software Development Kits, testing and experimentation.

3.1 The main idea

I wanted to create a somewhat realistic garden which is to show the potential of a VR headset. The main reason why I decided to make a garden was because of games like Stardew Valley (which is a farming simulator where the player can grow your crops and tend to your farm animals) and I wanted to create an experience that was similar to it.

Figure 3.1: Stardew Valley[1]

It was to be a place where you can just turn off and have some fun with your plants. As for the structure I was aiming at making a framework that is easy to extend and use if more plants or tools are to be added in the future.

[1] https://www.stardewvalley.net/

3.2 Basic needs of the plants

What are needed to grow a plant in real life? Basically water, sun, sufficient fertilization and to not get infected by insects. There are more than that, but those were the things I kept in mind when implementing my framework.

3.3 Prefabs(? más elnevezés ?)

3.3.1 Land

We have several lands which hold the script to check for water, fertilization and the insects themselves. When the game is started it calls the functions I created so that the land will dry out, will eat the fertilizer and call insects on itself and repeats calling it every now and then so the player will need to keep an eye out for the information shown in the User Interface for each land. For this I have used the `InvokeRepeating` (Invokes the method methodName in time seconds, then repeatedly every repeatRate seconds[2]) function of Unity which got placed inside the Start function, that runs when the player starts the game.

```
void Start() {
  /* Calls the dryOut function after 120 seconds
  ** then every 5 seconds */
  InvokeRepeating("dryOut", 120.0f, 5.0f);
}
```

The land prefab has a collider on it just like the tools so that is how you interact with it. Holds a reference to the plant Game Object which is null until you buy a plant through the User Interface. The User Interface shows the player information that given land is holding about water, fertilization and the insects. Water and fertilization can be an integer value between 0 and 100 while insects is a boolean whether is your plant infected or not which gets set in the `callInsects` method that will random a value and set it based on that.

[2] `https://docs.unity3d.com/ScriptReference/MonoBehaviour.InvokeRepeating.html`

3.4 Pole

The pole serves the sole purpose of buying and deleting your plant. When you touch the big red ball on the top of it with a controller it brings up the User Interface. For an object to be interactable I needed to use given VRTK scripts on it. The first script was the `VRTK_Interactable Object`, with its help I was able to edit whether the player can grab the given Game Object or use it or both. There are other options like a button needs to be held for usage or that the given object can only be used when grabbed.
Given that a controller is touching the sphere on the top, it bring up said user interface for buying and deleting the plant. Interaction with the UI was implemented using buttons and another VRTK script, named `VRTK_UI_Canvas`. It handles interaction with the controller pointer, so when the player points on a button and pulls the trigger, a given `onClick()` function will be called. The script that handles the User Interface functions for the pole is called `PoleUIController`. First when you touch the sphere the `PoleUIPicker` function gets called which checks if there is a plant already on the given land, if so then shows the UI for deleting the given plant, otherwise the player will see the UI to buy a new plant.
The UI for buying a brand new plant holds separate buttons for all the available ones with their names on it. When you click a button, it will instantiate that plant and set the plant for that land because by default the land holds a null as the plant. There was a small bug where the UI did not close after a transaction, so I set it to deactivate the windows after buying or deleting.
When the player decides to delete a plant that is already residing in the given land, it will delete that instance of the plant and set the plant to null on the land.

```
public void buyCatBush() {
  CloseUI();
  var bush = Instantiate(catBush, land.transform);
  bush.GetComponent<Grow>().setLand(land);
  land.setPlant(bush);
}
```

3.5 The plants

The plants basically build up of three different forms.
First one being when the player just purchased it and all it contains are bare branches. The model for it was made using Blender[3] which I cut up to smaller branches in Unity and joined them together with Fixed Joints. The sole purpose being so that the player is able to cut the branches off. In this phase of the plant, player needs extra attention because if the water, fertilizer levels are not sufficient and it has insects too, the game will start to rot the tree. Because this version of Unity supports getting the child of a component based on the hierarchy in the editor window, it was the base when implementing the rotting feature. The method itself is an implementation of a tree traversal on the branches. When given branch will not be cut down in a set amount of time then that branch will give the blight to its parent branches. If a branch rots instantaneously its children will start to rot too so this way I was able to make sure that a sub-tree with decaying root will be infected completely. This can be stopped via cutting said branches. In case we get through this phase, our plant can grow into its adult form, and at that point the rot will not endanger it anymore.
Second is when we have a mature tree. The base model in here too was made by Blender which gets switched from the branches. The player still needs to tend to the tree. I made a function that runs every few seconds until the plant is happy again so it can break out into blossoms.

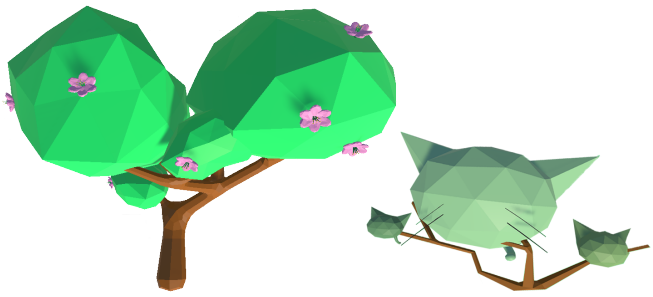


Figure 3.2: Available plants when they are in full blossom

[3] https://www.blender.org/

Chapter 4

Függelék

4.1 A program forráskódja

A függelékbe kerülhetnek a hosszú táblázatok, vagy mondjuk egy programlista:

```
while (ujkmodosito[i]<0)
{
   if (ujkmodosito[i]+kegyenletes[i]<0)
   {
      j=i+1;
      while (j<14)
      if (kegyenletes[i]+ujkmodosito[j]>-1) break;
      else j++;
      temp=ujkmodosito[j];
      for (l=i;l<j;l++) ujkmodosito[l+1]=ujkmodosito[l];
      ujkmodosito[i]=temp;
   }
   i++;
}
```

Nyilatkozat

Alulírott ..... szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet ..... Tanszékén készítettem, ... diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat / diplomamunkámat a Szegedi Tudományegyetem Informatikai Intézet könyvtárában, a helyben olvasható könyvek között helyezik el.

Szeged, April 17, 2019 .....

aláírás

Alulírott ..... szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet ..... Tanszékén készítettem, ... diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat / diplomamunkámat a TVSZ 4. sz. mel-

lékletében leírtak szerint kezelik.

Szeged, April 17, 2019 . . . . . .

aláírás

Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani **X. Y-nak** ezért és ezért . . .

Bibliography

[1] J. L. Gischer, The equational theory of pomsets. *Theoret. Comput. Sci.*, **61**(1988), 199–224.

[2] J.-E. Pin, *Varieties of Formal Languages*, Plenum Publishing Corp., New York, 1986.