CS CAPSTONE TECHNOLOGY REVIEW

DECEMBER 20, 2017

ANCESTRY DATA VIEWER

PREPARED FOR

ASHLEY MCGRATH

Signature Date

PREPARED BY

GROUP 22

TEAM ANCESTRY DATA VIEWER(ADVR)

MONICA SEK

Signature Date

Abstract

This document examines three pieces of technology that are involved in the creation of our project. Each piece will examine three possible technologies that can be used. After comparing and contrasting the options, a decision will be made to utilize one of the technology for the piece.

Contents

1 Introduction

The goal of this project is to take a GEDCOM file as an input and to display an organized ancestry tree as an output.

2 2D Visualization

2.1 Overview

For the first piece, we will be examining different options that can be used to display our program in its 2D application.

2.2 Criteria

The application needs to be able to display all the members required information and to clearly represent the relationship between members. The engine should be easy to manipulate so that the information is neat and not cluttered, even when the user is shifting the tree. The engine should have a VR-compatible feature that can be activated from desktop mode.

2.3 Potential Choices

The potential choices for displaying our project are the Unreal Engine 4, Unity Engine 3, and Lumberyard.

2.3.1 Unreal Engine

The Unreal Engine is a popular VR engines and is known for its ease of use. Switching from desktop to VR mode is also supported on this engine, there are published open-sources for this feature included [1]. The mode switching open-source code is among many other open-source data and an abundant supply of tutorial videos can be found throughout the community for help [2]. The Unreal Engine recently launched an enterprise team to focus on non-video game applications [3]. One of the target uses include data visualization. The software has over 20 built-in CAD data plugins that include features like geometry, texture, and camera angles [4]. Other parameter tools include uniform scaling and tiling awareness with objects.

2.3.2 Unity

The Unity game engine is another widely popular software for VR. This engine provides a built-in VR support that can be enabled from desktop [5]. Unity has also built support for non-video games application which focuses on architecture, engineering, and construction [6]. These tools can be used to display desired graphics and visualizations. Objects can be created in both 2D and 3D, and they can have specific details attached to them. Details are done by creating prefabs for the object, which will establish certain properties to each replicated object [5].

2.3.3 Lumberyard

The Lumberyard engine is a newer engine that was released by Amazon, it is still in its Beta version. It is also encouraged to be used for architecture, simulations, and animated movies purposes [7]. Lumberyard includes features like color grading, shading, particle effects, depth of field, etc. Free open source code is also available for the developer, that way they can just focus on changing aspects they want to change. Lumberyard also includes the concept of entities, which allows developers to drag and drop components to build the behavior they wish to see [7]. These components can be grouped together into slices to build more complex entities.

2.4 Discussion

All three technologies share open-source data that can be used to build functions for the application and can handle toggling to a VR application from desktop. Although they all are primarily game engines, they do provide features for non-video game application. Visual graphics are available in each engine however Lumberyard focuses on it the most and can deliver hyper realistic graphics. Unreal Engine and Unity still provide visual features that can be altered such as color and texture. Lumberyard and Unity utilizes grouping features that allows the developer to control properties across entities. Unreal Engine offers plug-ins and tools that can be used to alter entities.

2.5 Conclusion

As a team, we have chosen to use Unreal Engine because of its ease of use and overall responsive community of users which is beneficial for debugging issues. Although the Unity and Lumberyard provide complex handling with entities, our application desires minimal display that can be achieved with Unreal Engine and its built-in plugins. Unreal Engine also provides a larger option for non-video game applications after the launch of the enterprise community.

3 3D Visualization

3.1 Overview

For the next piece, we will be examining different options that can be used to display our program in its 3D application.

3.2 Criteria

The application needs to be able to display all the members required information and to clearly represent the relationship between members. The engine should be easy to manipulate so that the information is neat and not cluttered, even when the user is shifting the tree. The user should be able to look around the tree to examine neighboring nodes.

3.3 Potential Choices

The potential choices for displaying our project are the Unreal Engine 4, Unity 3, and Lumberyard.

3.3.1 Unreal Engine

The Unreal Engine is a popular VR engines and is known for its ease of use. A VR camera setup is included in the software which allows the developer to initialize camera view origins [2]. Depending on how the user should be oriented when using the program, the developer can adjust the initial camera setting. Unreal has its own VR measure units that can be converted to real-world units, so size is more manageable for the VR experience [2]. There are many open-source data and an abundant supply of tutorial videos that can be found throughout the community for help.

3.3.2 Unity

The Unity game engine is another widely popular software for VR. However, camera movement is also very limited [8]. Unity doesnt allow for direct VR camera change, for this to happen a change in position and rotation must be applied to a different object that is the parent of the object. The software also comes with a large option of built-in image effects including Colorful, Chromatica, Amplify Color, etc. Unity focuses on pixel ratio over lens correction, ultimately trading in performance for sharpness [5]. 3D objects also have specific graphic settings that also involves texture and color.

3.3.3 Lumberyard

The Lumberyard engine is a newer engine that was released by Amazon, that is still in its Beta version. High graphics and realistic visuals are highly supported by this software. Lumberyard utilizes gems to enable VR view [7]. Flow Graph nodes are used to control and add properties to the entities being used in the program [9]. The nodes can be used camera angle, image display, and controlling entities.

3.4 Discussion

All three engines can display 3D objects and utilize cameras. Unreal Engine is much more user-friendly with camera orientation. Lumberyard would be the next easiest to use while Unity is the more obvious limited one. Lumberyard and Unity provide a lot of graphic features that can be used to edit entities. Unity even sacrifices performance to be able to enhance sharpness of imagery. .

3.5 Conclusion

As a team, we have chosen to use Unreal Engine because its camera motion view has more options unlike the Unity engine which is much more limited. Lumberyard and Unity are ideal for high quality and complex graphics. Unity also consists of low performance which is something we dont want to sacrifice because we are using simple images.

4 VR API

4.1 Overview

For the final piece, we will be examining different options that can be used to configure the interface between our application and its controls.

4.2 Criteria

The API needs to be compatible with VR software and hardware. It should be capable of handling motion controllers to drag objects around in VR. Basic button configuration should be easy to configure.

4.3 Potential Choices

The potential choices for displaying our project are the Unreal Engine 4, Unity Engine 3, and OpenXR.

4.3.1 Unreal Engine

The Unreal Engine is one of the most popular VR engines and is known for its ease of use. It supports devices and contains built-in plugins to accommodate features [10]. The engine is compatible with a large amount of hardware including the Oculus Rift, PlayStation VR, Steam VR, Google VR, etc. Motion remote controllers setup is built into the software that allows for simple buttons and actions configuration. Events are set up through a blueprint interface displayed in a flowchart-fashion, the structure is simple and user-friendly while still having the ability to perform advance tasks [2]. Once the blueprint events are set up, there is a simulation feature to run tests.

4.3.2 Unity

The Unity game engine is another widely popular software for VR. It is compatible with the following hardware: Oculus Rift, PlayStation VR, Steam VR, Google VR, Gear VR, and Microsoft HaloLens. There are built-in APIs variables that can be used to detect the device being in use [**?**]. In addition, there are other script features available for API specifically. Images in VR are being provided twice, one for each eye, resulting in a performance cost [5].

4.3.3 OpenXR

OpenXR is a cross-platform VR API that connects a VR application from any VR system to access any VR device. This software eliminates the limited access between application and devices. The API is split into two levels of interface: application and device layer [11]. The application layer takes in events and outputs images to display. The device layer takes in controller state and outputs movement. These layers of translation are used to make the VR setup more uniform and available across multiple platforms.

4.4 Discussion

Unreal Engine, Unity and OpenXR support VR and motion remote controllers. In addition, both Unreal Engine and Unity can support a desktop to VR mode. Unreal Engines blueprint features allow for easier configuration by implementing flowcharts, and utilizing drag and drop actions. Unlike Unity which implements the configuration with code, because Unity is heavily code-based. OpenXR is an excellent option for integrating cross-platforms.

4.5 Conclusion

As a team, we have chosen to use Unreal Engine because it provides a simpler method to initialize the motion remote controllers. This feature is important to us because we want to incorporate these controllers into our program. Although OpenXR would be useful if we had integrated separate applications and devices, we are planning on keeping the hardware and application interaction simple. Unity also has a performance cost to incorporate VR images, which isnt favored for our application so Unreal Engine became the better decision.

REFERENCES

[1] DreaM&,
`https://answers.unrealengine.com/questions/262396/switch-on-vr-mode-in-runtime.html`

[2] Epic Games Inc,
`https://docs.unrealengine.com/latest/INT/Platforms/VR/`

[3] Ronnie Dungan,
`https://www.seriousgamesindustry.com/wp-content/cache/wp-rocket/seriousgameindustry.com/en/qa-simon-jones-directo`

[4] Randall Newton,
`http://gfxspeak.com/2017/08/22/reveals-enterprise-visualization/`

[5] Unity Technologies,
`https://unity3d.com/learn/tutorials/topics/virtual-reality/vr-overview`

[6] Unity Technolofies,
`https://store.unity.com/industries/aec`

[7] Amazon Web Services Inc,
`https://aws.amazon.com/lumberyard/details/`

[8] Unity Technologies,
`https://docs.unity3d.com/Manual/VROverview.html`

[9] Amazon Web Services Inc,
`http://docs.aws.amazon.com/lumberyard/latest/userguide/virtual-reality.html`

[10] Tom Looman
`http://www.tomlooman.com/getting-started-with-vr/`

[11] The Khronos Group Inc,
`https://www.khronos.org/openxr`