Tech Review for Noctilucent VR - Team 53

Taylor Fahlman

November 13, 2016

1 Introduction

I (Taylor), am in charge of the selection and implementation of the software component of the project. Regardless of the software used for viewing data, some framework needs to be used to integrate the Virtual Reality component. The three main frameworks in use today are the Open Source Virtual Reality framework, the Oculus Rift framework, and the HTC Vive framework. There also needs to be a framework for the UI. This is highly dependent on the software we select to view the data, but the benefits of each will still need to be explored.

2 Virtual Reality Development Framework

2.1 Options

The three options for the VR development framework are Open Source Virtual Reality, the Oculus Rift SDK, and the HTC Vive SteamVR SDK.

2.2 Goals

The goal is for the framework is to enable as many users of VR as possible, as well as to work with the software we choose for the data analysis.

2.3 Criteria

- Cross-platform - Well supported - Supports software chosen - Extensible

2.4 OSVR

The Open Source Virtual Reality project has two major components: a headset (hardware) and a framework (software) to enable other software to use various headsets. Here, the focus in on the viability of the software component of OSVR. One big benefit to using the OSVR framework comes from the title; open source. This means that the source code of OSVR is open and free for anyone to use, change, and contribute to. Specifically for our project, this is helpful as there could arise some limitation or feature that is needed that other frameworks do not provide. In that case, writing our own code of top of OSVR, and possibly integrating it back into the core is easy to do. Another benefit is that it works with a variety of headsets, and it is supposedly easy to add support for new headsets. The biggest downside is that while there is an official headset for OSVR, it is still not the most robust framework. There are many improvements that can be made still, and the lower overall quality of this technology shows in demos of both the software and hardware. Specifically for the software, just getting a demo to work was complicated. The firmware also needed to be upgraded, which is a technical and long process, and there have been reports of this bricking various hardware devices. As well, it does not support, for example, the Oculus Rift as well as the Oculus SDK does.

2.5 Oculus Rift SDK

The Oculus Rift SDK is a software development kit intended for the Oculus Rift headset. The main benefits of this software is that it is professionally made and supported, with a very high standard. It also is integratable with software such as the Unreal game engine, and the Unity game engine. It also has a large community to learn from and share with. However, there are several drawbacks. First of all, this is designed for gaming applications mostly. While most VR is as of today, there exists some limitations in the library for the kind of data analysis required of this project. The software that the Oculus already works with are not ones this project is considering, so this does not give the SDK an advantage per say. The largest drawback, however, is that this SDK is only targeted and provides official support for the Oculus Rift. Other headsets would be difficult or impossible to use with the Oculus Rift SDK. Due to the constraints and scope of the Oculus Rift SDK, it does not seem like the best choice for the project.

2.6 HTC Vive SteamVR SDK

The HTC Vive SteamVR SDK is also a software development kit, centered around the HTC Vive headset and SteamVR by Valve. The SDK is part of the larger HTC OpenSense SDK. The SDK is intended to be used with the HTC Vive headset, and with SteamVR, which is primarily focused at gaming. The SDK has many of the same advantages and disadvantages as the Oculus Rift SDK. It is professionally made and supported, has a relatively large community, and works out of the box with Steam, a very popular and widely used software distribution service. However, Steam almost exclusively distributes game software. While there are some utility and data analysis packages available, Steam and even more so SteamVR are focused on games. This again means that the library has some limitations for what the project is attempting. As well, it is aimed at only using the HTC Vive headset. Because of these constraints, this does not seem like the best choice for the project.

2.7 Discussion

Based on the criteria, OSVR meets all of them. It is cross platform, both operating systmes, and various headsets. It also supports any software, and is infinitely extensible. The only criteria where the other two have an advantage is in well supported. This is not ot say that OSVR is not well supported, but it doesn't have the same level or quality of support the others do. However, it is reasonably well supported and beats the other two options in the other criteria. Therefore, this project with use OSVR.

3 UI Framework

The UI selection is dependent on the software that is selected to analysze the data. Still, the following analysis takes into account the pros and cons of the technologies independent of the software used, and will recommend one that is the best for the job, even if it does not match the software we select.

3.1 Options

The three options explored here for the UI framework will be Qt, an HTML5/CSS/Javascript combination, and GTK+.

3.2 Goals

The goal is to find an easy-to-use and effecient UI framework. The selection also needs to be able to work as an overlay in a virutal reality space, or be easily extensible if not.

3.3 Criteria

- Easy to use - Well supported - Can be used with VR.

3.4 Qt

Qt is a popular UI framework. It basic components comprise of a C++ API, a seperate library, Qt Quick, in a language called QML, and a full IDE for Qt development. It is designed to work with large C++ applications as a UI layer on top of logic of the software. The general idea is that the software is written from the ground up integrated with the C++ API, or built using C++ and then using Qt Quick or some other form of Qt on top of it. Qt has many benefits. Primarily, it is widely used, with a great deal of support, and a long history of use spanning about 20 years. It is very versatile and is considered a standard UI framework by many. In addition to the C++ API and the Qt Quick, it also has HTML5 support for web development. Another benefit is that its cross platform, ready for Windows, Mac OSX, iOS, Linux and Android, along with some other less used target platforms. The downsides of Qt stem from some of its benefits. It is a rather large and complex system, and can be heavy computationally. There is a lot going on in a Qt application, and in the case of working with a pre-existing codebase, it may be difficult to display the data without breaking another part. Also, while it is likely easy to integrate as an overlay with VR because it is based in C++, there is no known official VR overlay support.

3.5 HTML5/CSS/Javascript

This combination of these web technologies is the standard of the modern web. This choice works well in the case we use a WebGL based software package. The benefits are versatility and near universal use. The shared online knowledge and resources for this choice are larger than arguably any other technology that could be considered here. The versatility of these technologies also mean that this selection is essentially unlimited in what it can do. There are many visualization tools built into Javascript in particular that would be useful. The main draw back is that with the size of the data this project will use, the limitations of these technologies would definitely show. There are known bottlenecks with complex graphics and these web techs. Also, there is no inherent support for overlay on a virtual reality HUD, so that will be an added challenge.

3.6 GTK+

3.7 Discussion