Virtual Reality and Motion Capture on the Web

Dan Erikson
2015

Bachelor of Science in Engineering Technology
Computer Game Programming

Luleå University of Technology
Department of Computer Science, Electrical and Space Engineering
Abstract

The capabilities of web browsers are quickly becoming more advanced and there are many new areas that still needs testing. This thesis is focusing on the functionality of virtual reality and motion capture in web browsers to determine their usefulness. To accomplish this a car simulator was built as a 3D web application with javascript to support for both technologies. The user is able to interact with the world using both the solutions at the same time or individually.

Sammanfattning

Funktionerna i webbläsare blir snabbt mer avancerade och det finns många nya områden som fortfarande behöver testas. Denna rapport fokuserar på funktionaliteten hos virtuell verklighet och motion capture i webbläsare för att bestämma deras användbarhet. För att åstadkomma detta så har en bil simulatork byggts som en 3D-webbapplikation med hjälp av javascript för att stödabåda teknikerna. Användaren kan interagera med världen genom att använda båda lösningarna samtidigt eller individuellt.
Acknowledgements

I would like to thank North Kingdom for providing me with the Gear needed for my thesis and even buying a driving wheel specifically for the project. Also thanks to Hans Eklund for testing and coming with input and suggestions.
# Contents

1  **Introduction**  
   1.1  **Goal and Purpose** .......................... 1

2  **Background and related work**  
   2.1  **HTML** ........................................ 2
   2.2  **JavaScript**  
   2.2.1  **WebVR** ...................................... 2
   2.2.2  **Three.js** .................................. 2
   2.2.3  **Leap.js** .................................... 3

3  **Methods**  
   3.1  **Development tools** .......................... 4
   3.2  **Social, Ethical and Environmental Considerations** 4

4  **Virtual Reality and Motion Capture Technologies**  
   4.1  **Head Mounted Display** ....................... 5
   4.2  **Position Tracking Camera** .................... 5
   4.3  **Motion detection** ............................ 6

5  **Result**  
   5.1  **Gameplay** .................................... 8
   5.2  **Virtual Reality**  
   5.2.1  **Cybersickness** ............................. 9
   5.3  **Input methods**  
   5.3.1  **Motion Capture** ............................ 9
   5.3.2  **Driving wheel gamepad** ................... 9
   5.4  **Controls** .................................... 10
   5.5  **Limitations** .................................. 11

6  **Discussion** ....................................... 12

7  **Conclusion** ....................................... 13

8  **References** ....................................... 14
1 Introduction

Virtual reality (VR) [1] is a computer-simulated environment that can simulate physical presence in places in the real world or imagined worlds. Only recently has the interest for it spiked thanks to new technology and hardware such as Oculus Rift [2]. Oculus Rift utilizes a head mounted display (HMD) [3] to keep track of the rotation of the head and the newest version also provides a camera to capture the position of the head. Oculus Rift has received support in many new games and simulators [4] even if it has only been released as development kits. It has also received experimental support for use in web browser, where Chrome [5] and Firefox [6] are supported at this time.

Motion capture (MoCap) [7] is the process of recording the movement of objects or people. It has many different uses for example film making and video game development. One of the devices used for MoCap is called Leap Motion Controller [8]. It is a small USB device that you either put on the table in front of you or mount to a VR headset. Moving hands above or in front of it will capture the movements for use in different games/applications.

This thesis focus on using these two technologies together to test their capabilities in web browsers since there has only been a few projects using them [9][10][11]. Especially VR for use in web browsers needs testing since support for it only exists in test builds of both Chrome and Firefox. A driving simulator has been created to test these technologies. The user sits in the driving seat and can look around with the Oculus Rift and steer the car with the help of the Leap motion Controller.

The project has been developed at North Kingdom[1] who provided all the necessary tools needed to complete the goals of the project. North Kingdom is a company that creates projects for many other well known companies with very different goals. They are able to take use of the results achieved in this project if they get a project that requires use of VR or MoCap on the web.

1.1 Goal and Purpose

The goal for this project was to create a simple 3D environment where the user can move around to test the capabilities of VR and MoCap in web browsers.

1. How well does virtual reality and motion capture in web browsers compare to regular applications?

2. Are virtual reality and motion capture projects in web browsers ready to be deployed to regular users?

http://www.northkingdom.com/
2 Background and related work

VR is a very interesting technology since it can help to immerse the user into simulations especially with the help of other technologies such as MoCap. Something that has not been explored much is implementation of VR in web browsers which could help the user interact with the content. An example would be if a clothes shop supported VR, the user would be able to put on a 3D model of the clothes or look at the clothes from different angles.

The only current way to use Virtual reality in web browsers and gain access to the VR devices is through the javascript application programming interface(API) WebVR. To gain access the API we will use HTML5 in combination with the Javascript language. The javascript libraries Leap.js and Three.js are used to gain access to the Leap Motion Controller and easily output graphics through WebGL.

2.1 HTML

HTML is the standard markup language used to create web pages and is short for HyperText Markup Language. It is maintained by the World Wide Web Consortium(W3C) which is an international community that develops open standards to ensure the long-term growth of the Web. The latest version is HTML5 which was standardised in October 2014. HTML5 adds support for the latest multimedia and graphical content without having to resort to plugins and APIs.

2.2 JavaScript

JavaScript is the programming language of HTML and the Web. HTML is used to show the content while javascript is used to allows client side scripts to interact with the user, control the browser and alter the document content that is displayed. Javascript is the only programming language supported by the major browsers and omits the need for plugins like Adobe Flash.

2.2.1 WebVR

WebVR is an experimental Javascript application programming interface(API) that provides access to VR devices. It is currently provided with experimental builds of Chrome and Firefox.

2.2.2 Three.js

Three.js is a library for javascript that makes WebGL easy to use. WebGL is a web standard for a low level 3D graphics API and is visualized through the HTML5 canvas element. Three.js replace complex and long code when drawing and editing 3D graphics with simple functions and is provided with a detailed documentation. With the help of VRRenderer Three.js can take advantage of the upcoming WebVR API.
2.2.3 Leap.js

Leap.js is the javascript plugin framework that handles the Leap Motion Controller. There are plugins available that can handle input data from the Leap Motion Controller, draw a hand in a 3D scene using three.js that mimics the user’s movement, push buttons and drag planes.
3 Methods

To test VR and MoCap in web browsers a driving simulator was implemented. The user is able to steer the car around in a small confined city and look at the surroundings with the help of VR. MoCap takes the location input of the user’s hands and use it to turn the driving wheel. There were no elegant solution to handle acceleration with MoCap so the keyboard is used for that part. Other input methods were also tested separately to compare how they handle cybersickness. The other input methods are a regular keyboard and a driving wheel with supplied pedals hooked up to the computer.

3.1 Development tools

The project was developed in HTML and javascript with the help of Webstorm [23] which provides features to edit in run-time and break in the middle of the code to debug. This helped to speed up the development process and enabled fast debugging.

3.2 Social, Ethical and Environmental Considerations

Cybersickness is a common occurrence in VR and there are different theories [24] as to why this happens. The most widely accepted theory is the sensory conflict theory in which it is believed that inconsistency between senses that gather information about the body’s orientation and motion causes a conflict that the body does not know how to handle which will then result in cybersickness. This was taken into account during the development of the project and had some impact on the final result.
4 Virtual Reality and Motion Capture Technologies

Virtual reality tries to simulate a computer created environment in which it can recreate different sensory experiences. It can be used for training purposes to prepare for real life scenarios or for pure entertainment. The most commonly used device that most people identify with VR are HMDs. There also exist different input methods that help the user interact with the environment like a glove [25] that capture your hand movements or cameras [26] that can capture different points from your body and use them for movement.

4.1 Head Mounted Display

HMDs, as shown in figure 1, are used to always keep the display in front of the user no matter where they might look. They can then be modified to also handle user input such as the rotation of the head to further immerse the user into the virtual experience. HDMs also use a separate screen for each eye which create a 3D effect.

![Figure 1: Oculus Rift dev 2.](image)

4.2 Position Tracking Camera

A position tracking camera, as shown in figure 2, can be used in combination with HMDs to keep track of the head position. This will add another layer to the VR experience since the user will be able to move around his/her head with actual physical movements and will also result in reduced cybersickness.

![Figure 2: Position Tracking Camera.](image)
4.3 Motion detection

Motion detection is used to capture movement of a real object and translate it into computer data. Older solutions used markers to record different spots as seen in figure 3. The data is collected from every marker attached to the actor and then used to animate the skeleton that is attached to the actual model. The method used in this project take advantage of markerless motion capture, as shown in figure 4 in which advanced computer vision technology and algorithms are used to achieve real time motion detection so the actor won’t have to wear anything.
Figure 4: Leap Motion in use.
5 Result

The goals for this project were successfully completed as first envisioned. A drivable car with support for different input methods has been implemented. The 3D environment does not use state of the art graphics but does fulfill its purpose to simulate a small confined city in which the user can drive around.

There were no major setback during implementation of the project since all the APIs functioned well and were well documented.

The results gathered here are from my own observations, but some inputs have been given from different users that tested the project during the development. The input was mostly concerning cybersickness and small input modifications.

5.1 Gameplay

The project is similar to most driving games. The user sits inside a car and can drive it around in the confined area. There is no real objective so the project is more like a simulator than an actual driving game.

![Data from both the Oculus Rift HMD and position tracking camera used for in game camera movements.](image)

5.2 Virtual Reality

VR was successfully implemented with the help of WebVR and the Oculus Rift HMD. There are two different modes for displaying data to the Oculus Rift. The first is direct HMD access mode in which the application takes direct control
over the HMD to display images. The second is extended desktop mode in which the HMD is regarded as another monitor and has the desktop extended to it. The direct access mode is more stable since it can have a direct connection to the application while the extended desktop mode provides a choppier experience which in turn induces cybersickness.

5.2.1 Cybersickness

When first implementing VR into the project the cybersickness was very noticeable even if the user was just sitting still and looking around. After successfully implementing the position tracking camera that accompanies the Oculus Rift, the effects of cybersickness was significantly reduced. The camera allows the user to move his/her head around freely in the car and look at things from different angles as seen in figure 5. This adds another layer of realism to the implementation and helps the senses to feel more in sync with each other.

5.3 Input methods

There are three different methods for providing input to the application:

- Keyboard
- Leap Motion
- Driving wheel gamepad

The main goal was to test how well the motion capture from the leap motion controller worked, but at the same time the other methods were compared with it to test which method worked best in combination with VR.

5.3.1 Motion Capture

The Leap motion controller works as well in web browsers as it does in regular computer programs, however the controller is not as accurate as one would expect. The controller can only see one side of the hand at a time, so if fingers overlap or are obstructed by the palm they may not register correctly. The implementation did have support to use both hands at the same time, but one hand is still needed to accelerate and decelerate using the keyboard. The Leap motion does not provide any physical response while touching things so it can become the cause of some conflicts between the senses.

5.3.2 Driving wheel gamepad

The driving wheel gamepad works great in combination with VR since it gives something physical to hold onto while looking around which can help to reduce cybersickness. The gamepad API worked fine with both Chrome and Firefox but the location to get some of the axes and buttons were different depending on the browser. The version of Chrome used in the project has a bug if you
refreshed the page or switched tab, it would forget all the connected gamepads until Chrome is restarted.

Figure 6: Visualisation of Leap motion in use.

5.4 Controls

The controls only handles turning and moving backward or forward which are the basic controls for a car. The vehicle needs to be in motion to be able to turn and the rotation is based on the speed of the vehicle. The vehicle also simulates friction by decreasing speed over time when the user is not accelerating forward or backward.

Keyboard is the simplest method where the only keys used are:

- W or Up key - forward
- S or Down key - backward
- A or Left key - turn left
- D or right key - turn right

Leap Motion only handles the turning of the vehicle and still needs the keyboard to move backward or forward. When a hand is found by the Leap Motion controller a representation of it will be shown somewhere around the driving wheel as seen in Figure 6. The user needs to do a grabbing motion or pinch two fingers together for the wheel to react to input, then move the hand from left to right as one would do while driving a real car.
The driving wheel gamepad uses a wheel to turn and pedals to move. The gas pedal is used to move forward and the clutch is used to move backward.

### 5.5 Limitations

When you create a 3D web application you must take into account that it is not as powerful as a native application and be careful with the assets used when implementing it. Either the assets needs to be properly optimized to be able to include a significant amount or there should only be a few of them so it does not impact the frame rate or cause a crash.
6 Discussion

The development time for the project was 2.5 months so it might not have been enough time to locate all the flaws VR and MoCap within a web application. The impression obtained from the whole process is that both VR and MoCap have at least come a long way and any major problems or bugs have been taken care of.

VR in web browsers does have its problems compared to regular applications. At the moment it is not very user friendly since specific builds of web browsers have to be downloaded to access the feature. This will limit the content to those who are more familiar with webVR, however no bugs or other issues were noticed while working with the web implementation for Chrome so if it would be released to the live build it would not be an issue. The Firefox version still needs some updates since it does not have support for the direct HMD access mode which is a very important feature since the extended desktop mode had some choppy results while looking around.

The Leap motion controller had the same result in web browsers as it had in native applications but the controller itself can often be very inaccurate and inconsistent. There is not much that can be done about this except to upgrade the actual hardware to something that can easily follow finger movements even though they may be in strange positions and capture what is happening around the whole hand instead of just one side that is currently limiting the Leap Motion controllers usefulness.

The Driving wheel gamepad controller is a perfect replacement for the motion controller since it provided a more realistic feel and could prevent cybersickness since it provides real feedback both on the hands while holding the driving wheel and on the feet when using the pedals. This is however only true for specifically driving games/simulators since the controller is made with the parts of a car in mind. Other games would need to utilize controllers specifically built for their type of input for it to work in a similar way. An example of another controller would be a gun controller made specifically for shooting games, however this would not be enough to provide a equally immersing feeling since the user still needs to move around in the virtual world while he/she would still be standing or sitting still in the real world.
7 Conclusion

The conclusions of this thesis is mostly from the developers perspective but does have some input from other users that briefly tested the solution.

Most of the time dedicated to research was used to create a web application with the intent to test different input and output combinations of VR and MoCap. The user sits inside a car which is located in a small confined city that he/she has control over.

1. How well does virtual reality and motion capture in web browsers compare to regular applications?

The VR implementation in web browsers is still in the beta stage but it has come a long way. Chrome has the best integration at the time with support for direct HMD access mode which will provide the most seamless experience and wont require you to fumble around with window buttons you cant see inside the Oculus Rift.

The Leap motion implementation works equally well in both browsers and native apps both is in general a poor choice when it comes to accuracy and consistency.

2. Are virtual reality and motion capture projects in web browsers ready to be deployed to regular users?

At the current time this is not an option since the Oculus rift has not yet been released as a user product, only as a developer unit. The web VR implementation is also in a beta stage which could prove to be too much of a hassle to set up for the user.

The Leap Motion web implementation did work but it had some problems with different hand motions which could annoy the user and provide a problem where accuracy is needed. For this to work there would need to be an update to the actual hardware of the Leap Motion controller.
8 References

References

http://www.cybertherapy.info/pages/telepresence.pdf

https://www.oculus.com/dk2/

[3] Ivan E. Sutherland, “A head-mounted three dimensional display”, 1968,
http://design.osu.edu/carlson/history/PDFs/p757-sutherland.pdf


http://blog.tojicode.com/2014/07/bringing-vr-to-chrome.html

http://mozvr.com/downloads/

[7] Organic Motion, “What is Motion Capture?”, 2015,
http://www.organicmotion.com/motion-capture/

[8] Leap Motion, “Leap Motion Controller”, 2015,
https://www.leapmotion.com/product

[9] Pehrlich, “VRCollage”, 2015,
https://developer.leapmotion.com/gallery/vrcollage

[10] Pehrlich, “VR Quickstart”, 2015,
https://developer.leapmotion.com/gallery/vr-quickstart

https://developer.leapmotion.com/gallery/light-pinch-vr

http://www.w3schools.com/js/js_intro.asp

http://webvr.info/

http://www.w3schools.com/html/html5_intro.asp

[15] Leap Motion, “Welcome to LeapJS”, 2015, 2015,
http://blog.leapmotion.com/leap-js-easy-leap-motion-functionality-with-javascript/
[16] Three.js “What is Three.js”, 2015,
http://threejs.org/docs/index.html#Manual/Introduction/Creating_a_scene

https://www.khronos.org/webgl/

http://www.w3.org/html/

http://www.w3.org/Consortium/mission.html


http://tyrovr.com/2014/06/29/three-vr-renderer.html

[22] Leap Motion, “Plugins”, 2015,
https://developer.leapmotion.com/downloads/javascript

https://www.jetbrains.com/webstorm/


[25] CyberGlove Systems, ”CyberGlove III”, 2015,
http://www.cyberglovesystems.com/?q=products/cyberglove-III/overview

[26] OptiTrack, ”Flex 3”, 2015,