Final Year Project

Immersive telepresence System v1

Prakash Waghela

2017

# Acknowledgements

# Abstract

Table of Contents

[Chapter 1 - - Introduction 5](#_Toc481063260)

[1.1 Project Summary 5](#_Toc481063261)

[1.2 Literature Review 5](#_Toc481063262)

[1.2.1 Telepresence 5](#_Toc481063263)

[1.2.2 Robotics 5](#_Toc481063264)

[1.2.3 Gesture Control 5](#_Toc481063265)

[1.3 Gap Analysis 5](#_Toc481063266)

[1.4 Technical Review 5](#_Toc481063267)

[1.4.1 Myo Armband 5](#_Toc481063268)

[1.4.2 Leap Motion 6](#_Toc481063269)

[1.4.3 Oculus Rift (DK2) 7](#_Toc481063270)

[1.4.4 Nao Robot 8](#_Toc481063271)

[Chapter 2 - Solution Design 9](#_Toc481063272)

[2.1 Aims and Objectives 9](#_Toc481063273)

[2.1.1 Functional Objectives 9](#_Toc481063274)

[2.1.2 System Objectives 9](#_Toc481063275)

[2.2 Project Plan 9](#_Toc481063276)

[2.2.1 Phase 1 - Input Devices 9](#_Toc481063277)

[2.2.2 Phase 2 - Output Devices 9](#_Toc481063278)

[2.2.3 Phase 3 - Final Solution 9](#_Toc481063279)

[Chapter 3 - Implementation 10](#_Toc481063280)

[3.1 Phase 1 - Input Devices 10](#_Toc481063281)

[3.1.1 Myo Armband 10](#_Toc481063282)

[3.1.2 Leap Motion 10](#_Toc481063283)

[3.1.3 Oculus Rift 10](#_Toc481063284)

[3.2 Phase 2 - Output Devices 10](#_Toc481063285)

[3.2.1 Oculus Rift 10](#_Toc481063286)

[3.2.2 Nao Robot 10](#_Toc481063287)

[3.3 Phase 3 - Final Solution 10](#_Toc481063288)

[Chapter 4 - Evaluation 11](#_Toc481063289)

[4.1 Testing 11](#_Toc481063290)

[Chapter 5 - Conclusion 12](#_Toc481063291)

[5.1 Future Work 12](#_Toc481063292)

[References 13](#_Toc481063293)

# - Introduction

## Project Summary

## Literature Review

In this section, I have explored concepts connected to the scope of this project, looking at relevant related work and discussing the significance in relation to my project.

### Telepresence

### Robotics

### Gesture Control

## Gap Analysis

## Technical Review

### Myo Armband

The Myo Armband (Thalmic Labs (a), 2017) is a wearable band that can be worn around the forearm of the user.

#### Sensors

* 8 Electromyography sensors
  + The EMG sensors can be used to detect the electrical signals in the muscle. These readings can be manipulated to detect specific hand gestures and motions.
  + The raw EMG readings can be captured from the Myo through the SDK with a single reading being available per sensor, at a rate of 200Hz (Bernhardt, 2015).
* Three-axis gyroscope
  + The gyroscope can be used to detect rotational movement of the arm. There are three planes in which an object can be rotated (roll, pitch, yaw), all of which can be measured by the sensor.
  + The data from the gyroscope can be captured through the SDK, with a single reading in degrees per second (Thalmic Labs (b), 2017) being available for each rotation axis at a rate of 50Hz (Bernhardt, 2015).
* Three-axis accelerometer
  + The accelerometer can be used to detect movement of the arm. In specific, the sensor can detect the change in velocity and position of the arm.
  + The data from the accelerometer can be captured from the Myo through the SDK, with a single reading in units of G (Thalmic Labs (b), 2017) being available per sensor at a rate of 50Hz (Bernhardt, 2015).
* Three-axis magnetometer
  + A magnetometer can be used to detect magnetic fields. This allows allow the armband to act as a compass, using the Earth’s magnetic field to determine the absolute orientation.

#### Interface

* Connectivity - The Myo armband has a Bluetooth interface that allows the user to connect the armband to a variety of compatible devices. This means the device can be operated without the use of wires, allowing for unrestricted movement while using the device.
* Battery - The Myo armband has a built-in battery which can be recharged through a micro USB port on the device. The battery is advertised to allow for “One full day use out of a single charge” (Thalmic Labs (a), 2017).
* Feedback - The Myo armband can provide direct feedback through two methods.
  + Haptic - The armband utilises short, medium, and long vibrations to notify the user of different events.
  + LEDs - The armband utilises LEDs on the device to notify the user of different device statuses.
* SDKs - Native SDK is available in C++.

#### Usability

The Myo armband is a lightweight and small device which easily fits around the forearm of the user.

* Gestures - The software for the Myo armband has five pre-programmed hand gestures (Thalmic Labs (a), 2017) that it can detect through the EMG sensors.
* Calibration - The software runs the user through a calibration step to create a user profile for the armband. This improves the reliability of the device and is specific to each user due to the variation in EMG readings through the arm of the user.

### Leap Motion

The Leap Motion is a small, bar-shaped motion detector. The device can be placed on a flat surface and comes packaged with software to detect and track the user’s hands.

#### Sensors

* Infrared Cameras - All data provided by the Leap Motion is derived from images taken by two infrared cameras contained within the base (Colgan, 2014), taken at up to 200 frames per second (Leap Motion Inc (a), 2017). The Leap software analyses the images and provides data from the motion of the tracked hands.
  + The cameras utilise wide angle lenses creating a large interaction space which is shaped like an inverted pyramid (Colgan, 2014).
  + The images are fed into the Leap Motion software which returns a 3D representation of the device’s field of view. The software can detect when a user’s hand is captured in an image and the software infers a large amount of data from the analysis of the image, allowing for the user’s hand to be tracked in 3D space.

#### Interface

* Connectivity - A single wire is required to connect the Leap Motion device to any USB port on a compatible device. The device is powered through the USB connection and does not require charging.
* Feedback - The Leap Motion has a green LED on the side of the panel to indicate that it is connected to a PC. When powered and connected, there are three red lights lit on the surface of the panel. All other feedback is given through the software.
* SDKs - Native SDKs are available in a wide variety of languages, including C++, Python, JavaScript, Java, and C#.

#### Usability

Once set on a surface, the Leap Motion is easy to control through the supplied software.

* Visualizer - Software to build a visualisation of any hands detected over the Leap Motion in real-time comes packaged with the software. This visualizer is useful for quickly testing whether a given hand motion is captured correctly by the device.
* Troubleshooting - Within software package is a troubleshooting page which allows the user to see any faults detected in the hardware at a glance.

### Oculus Rift (DK2)

The Oculus Rift DK2 is a head mounted virtual reality headset. The device is a developer’s kit and as such has lower specs than the consumer device that is available on the market.

#### Sensors

* The headset contains a gyroscope, accelerometer, and magnetometer (Oculus, 2017). The data from these sensors is combined with a model of a user’s neck and head to translate the real-time head movements made by the user into rotational data (pitch, yaw, roll) which can be sampled at up to 1000Hz (Oculus Blog, 2013).
* The headset contains an array of infrared LEDs which are tracked by the external camera supplied with the device (Oculus, 2017). The camera is used to track the LEDs, and as such track any movements in 3D space. This movement measurement can be sampled at 60Hz (VR&AR Wiki, 2017) (RiftInfo.com, 2016).
* The Leap Motion comes with an add-on mount which allows the user to slot the Leap Motion device onto the front of the Oculus Rift. The Leap Motion can then be utilised by tracking the user’s hand in front of them rather than above the Leap Motion on a surface.

#### Output

* The headset contains two lenses, with each lens able to display media with a resolution up to 960 x 1080 at a refresh rate of 75Hz. Due to the high graphic requirements, a powerful PC is required to utilise the output capabilities of the Oculus Rift.
  + Any image that is to be output using the Oculus Rift must be pre-processed to account for the distortion required for the image to be seen correctly.

#### Interface

* Connectivity - Utilises multiple cables and requires a high-performance PC to function.
  + Could possibly utilise a lower-performance PC if only the positional tracking was utilised as the graphics rendering is the cause of the high-performance PC requirement.
* SDK - Native SDK is available in C++.

#### Usability

The Oculus Rift is a device which requires you to be tethered to the PC with multiple wires.

* The headset requires a tight fit to create an immersive experience. This can be uncomfortable for some users.
* To utilise the tracking capabilities, you must remain within the view of the camera used to track 3D motion.
* Due to the multiple components and wires required to correctly operate the device, there are sometimes sync issues with one component not being recognised correctly.
* The Oculus requires a powerful PC with the following minimum specifications:
  + NVIDIA GTX 970
  + Intel i5-4590 or better
  + 8GB+ RAM
  + 1x HDMI, 3x USB (3.0)

### Nao Robot

The Nao robot is a humanoid robot that can be programmed to carry out a wide variety of tasks (Aldebaran Documentation, 2016).

#### Sensors

* Capacitive tactile sensors - The robot has tactile sensors installed on its head, hands, and feet. Along with these sensors is a button located on the chest.
* Gyroscope and Accelerometer - The robot has access to inertial data, centred around the robot’s torso.
* Sonar - The robot has sonar capabilities that allows it to estimate the distance from objects ahead.
* Microphone - The robot has microphones installed on its head that provide a method for sound to be captured.

#### Output

* Loudspeakers - The robot has speakers installed allowing it to broadcast sound.
* Video Camera - The robot has two cameras installed which is capable (in ideal conditions) of providing video of resolution up to 1280 x 960 at 30 frames per second.
* LEDs - The robot has a multitude of LEDs installed which are used to signify different robot states. The predominate LEDs are located on the head, in the eyes, ears, and within the chest button.
* Motion - The robot has 5 parts that make up the body of the robot, which utilises a total of 25 separate motors to allow for a wide array of movement.

#### Interface

* Connectivity - Connection to the robot is possible via an access point. The robot has support for either a wired ethernet connection or a wireless Wi-Fi connection.
* Choregraphe - A software suite available from the Aldebaran website which has an interface to view and broker connection details. Choregraphe also allowed the user to sample movements through a drag and drop interface.
* Virtual Robot - A virtual version of the robot with reduced functionality is made available through Choregraphe.
* Open Nao - The firmware installed on the robot which is based on a Linux environment. The local environment can be accessed through an SSH client.
* SDKs - Native SDKs are available in C++, Python, and Java.

# Solution Design

## Aims and Objectives

The over-arching aim of this project is to create an interface which can be utilised as a telepresence system. This aim can be split into the following objectives:

### Functional Objectives

1. The user will be able to control the movement of the robot in 3D space, using some gesture as the input.
2. The user will be able to control the movement of the head of the robot, using some gesture as the input.
3. The user will be able to control each arm of the robot individually. The arm should be able to be controlled to a level where the robot is able to interact with the environment.
4. The user will be able to utilise the speakers on the robot to relay some message.
5. The user should be able to get feedback from the robot, in the form of a camera image.
6. The user should be able to control the robot while it’s untethered.
7. The motions required to move the robot will be relatable to the motion made by the robot.
8. The movement of the robot should be controlled and measured; any erratic movements or latency should be minimised.

### System Objectives

1. The delivered project should be usable by any person with minimal training.
2. The delivered project should run on either Windows, Mac, or Ubuntu.
3. The delivered project may require a PC capable of utilising the Oculus Rift hardware.

## Project Plan

After reviewing the technology available for this project, I decided to split work into three distinct phases. In the first phase of work, I would examine the devices that could be used to gather input data. In the second phase of work, I would examine the devices that could be used to give some output to the user. In the third and last phase, I would look at combining work from the first two phases to create the final project deliverable.

### Phase 1 - Input Devices

### Phase 2 - Output Devices

### Phase 3 - Final Solution

# Implementation

## Phase 1 - Input Devices

### Myo Armband

### Leap Motion

### Oculus Rift

## Phase 2 - Output Devices

### Oculus Rift

### Nao Robot

## Phase 3 - Final Solution

# Evaluation

## Testing

# Conclusion

## Future Work

# References

Bernhardt, P. (2015, March 14). *Myo - Blogs*. Retrieved January 2017, from Myo: http://developerblog.myo.com/myocraft-logging-imu-and-raw-emg-data/

Colgan, A. (2014, August 9). *Leap Motion - Blogs*. Retrieved January 2017, from Leap Motion: http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/

Leap Motion Inc (a). (2017, January). *Leap Motion - Developer Docs (a)*. Retrieved January 2017, from Leap Motion: https://developer.leapmotion.com/documentation/javascript/api/Leap.Controller.html

Oculus Blog. (2013, January). *Building a Sensor for Low Latency VR*. Retrieved April 2017, from Oculus: https://www3.oculus.com/en-us/blog/building-a-sensor-for-low-latency-vr/

Thalmic Labs (a). (2017). *Myo - Technical Specs*. Retrieved January 2017, from Myo: https://www.myo.com/techspecs

Thalmic Labs (b). (2017). *Myo - Device Listeners*. Retrieved January 2017, from Myo: https://developer.thalmic.com/docs/api\_reference/platform/classmyo\_1\_1\_device\_listener.html

Thalmic Labs. (2017). *Myo - Technical Specs*. Retrieved January 2017, from Myo: https://www.myo.com/techspecs