The Tennis Trainer

Development of a Myo armband application

Bachelor Thesis for Creative Technology by

Kevin Vogelzang

Supervisor: Erik Faber

Critical Observer: Douwe Dresscher

19 July 2017

Abstract

The Myo armband by Thalmic Labs allow the user to control application by means of movement and gestures. These gestures are measured using EMG to measure the power exerted by the muscles. Furthermore, the movement is measured using an Inertial Measurement Unit. Currently quite some applications have been made for the Myo, however there is still enough room for the development of unique applications.

To explore what unique applications could be made for the Myo a research question was formulated, being: "In what domain(s) can the Myo provide actual benefit?".

This question has been answered with a state of the art research into the current applications for the Myo as well as a research into related technologies to the Myo. The application domains for the Myo are found to be Mouse and keyboard replacement, Virtual reality, Entertainment, Robotics and Health. Where Health consists both of Medical and Sports. The actual benefits provided by the Myo are identified to be handsfree usage, large range and battery life as compared to the LEAP Motion, Kinect, Fitbit and more.

The first research question provided the basis for the second research question: "What application can be built so that the Myo actually provides this benefit?". This research question has been answered using the Creative Technology Design Process which consists of an ideation, specification, realisation and evaluation phase.

The ideation phase resulted in three possible applications a guitar trainer, a remote smart screen control for teachers and a tennis trainer. using the unique aspects of the Myo and via interviews the choice was made to create the tennis trainer application.

The specification phase provided the basis for the application. Using a functional analysis, system architecture and a requirement analysis the application was given form. The most important requirements formulated here were the usage of the Myo, the visualisation of power and movement and storage of the workouts.

The realisation phase covers the creation of the final prototype. Before this prototype was made a set of Lo-Fi prototypes was created to explore certain functionalities of the final prototype. Afterwards the final prototype was created which implemented the requirements and system architecture formulated in the specification phase.

In the final phase, the evaluation phase, the Hi-Fi prototype was tested using user evaluations. 21 people with an age range of 18-25 years old participated within these user tests. The overall result was that the application was received in a positive manner with some specific points being that the choice for using a Myo as sensor input was found logical and not hindering. The average grade given by the participants was a 7.2.

Future development into the application would involve the improvements on the feedback of certain functionalities, such as the calibration and favourite feature. Furthermore, visualisation of the tennis movements can be made clearer. Also the workout names should be improved. Additional features of the application would include the addition of video footage, the recognition of swing types and qualitative feedback upon these swings.

Acknowledgements

There are several people I would like to specifically thank for helping me within this project.

First of all, I would like to thank Erik Faber, the supervisor of this project, who offered a lot of feedback and guidance throughout this project. The weekly meetings really helped me stay on track and allowed me to finish the various phases in the correct order instead of directly starting by creating prototypes.

Second, I would like to thank Douwe Dresscher, the critical observer of this graduation project. Although we did not meet, I would like to thank you for taking the time to read through and grade my bachelor thesis.

Finally, my thanks go out to all the students, friends and family who have helped me throughout this project by providing feedback, moral support and input for interviews and evaluations.

Content

1. Introduction

The Myo armband by Thalmic Labs on which this graduation project is based will be introduced in this chapter. Followed by the challenges regarding this project are listed. Subsequently, the research questions which will be answered are mentioned. Finally, the outline of the rest of this thesis is provided.

1.1 Myo armband

The Myo armband (see figure 1) is a wearable sensor created by the Canadian company Thalmic Labs [1]. When placed on the widest part of the forearm the Myo is able to measure the muscle activity, the movement and the orientation of the arm and hand. It does this by reading out the electromyography (EMG) of the user to measure the muscle activity as well as using an accelerometer, gyroscope and magnetometer to measure the movement. The Myo is able to recognise five preprogrammed gestures. A further explanation is provided in chapter 2.

Currently there have been some applications built for the Myo. Most of these applications use the aforementioned gestures, movement and orientation data to control the applications [2]. It is possible to create one's own applications using the Software Development Kit (SDK) provided by Thalmic Labs [3]. The applications which are currently available in the marketplace enables one to control various programs and hardware devices. These programs range from specifically created games to a controller for a presentation [2]. Some of the hardware devices that can be controlled with a Myo application are the Parrot drone and an Arduino [2].

The Myo is also used in research since it is a relatively cheap wearable sensor. A couple of examples of the research being done is to control a prosthetic arm [4] or to combine the Myo with virtual reality to create a greater immersion [5].

Figure 1: The Myo armband in use [6]

1.2 Challenges

The Myo does not have many applications and most of the applications serve as a keyboard replacement for an already existing program. This usually results in the Myo not actually adding any new functionality. Due to the unique feature of the Myo, it has a lot of potential which currently is not being used to its full potential.

The challenge of this project is to create an innovative application for the Myo. For this to be successful, research in what domain the Myo can be best used in has to be done first. Once the possible application domains are known an actual application for the Myo will be made which taps into this potential.

1.3 Research Questions

To specify the above mentioned challenges two research questions have been formulated. The first research question focuses on gaining insight into the current state of the art of the Myo and will provide insight in how to answer the second question. This research question is also supported by three subquestions.

- "RQ1: In what domain(s) can the Myo provide actual benefit?"
    - "SQ1.1: What are the current applications the Myo is being used for?"
    - "SQ1.2: What are possible applications for the Myo which currently do not exist?"
    - "SQ1.3: What is the added benefit the Myo provides?"

The second research question builds upon the first and will focus on the application that will be created as the final product.

- "RQ2: What application can be built so that the Myo actually provides this benefit?"

1.4 Structure of report

The structure of the rest of the report will be described here. The structure is based on the Creative Technology Design Process, which will be explained in the Method section.

Chapter 2 focuses on the State of the Art review on Myo applications. The goal here is to gain insight into the domains the Myo is currently used in in order to provide a basis for an answer to the first research question.

Chapter 3 focuses on the methods used within this project. The design process used for this project will be described here. The usage of this design process results in various other methods and techniques to be used, which are also further elaborated upon in this chapter.

Chapter 4 focuses on the ideation phase of the project. Here the idea generation based on the technology used, user needs and a creative idea for the project will be described. Furthermore the idea will be further elaborated upon, finalising in a set of preliminary requirements.

Chapter 5 focuses on the specification phase. Here the functional analysis, system architecture and final requirements are specified and discussed.

Chapter 6 focuses on the realisation phase, where first the creation of the Lo-Fi prototypes is described. After the discussion of the tests on these, the creation of the final Hi-Fi prototype is discussed.

Chapter 7 focuses on the evaluation of the final product. In here it is tested whether the set requirements are met by user evaluations.

Chapter 8 consists of the conclusion and recommendations of the project. Here the project is summarised and recommendations for future development are stated.

2. State of the Art review on Myo applications

In this chapter the State of the Art of the Myo and its applications are discussed. First, the Myo will be explained in a more elaborate way, this will be followed by a background research on the creation of a gesture recognition system. There after, the State of the Art research on current application domains and potential domains will be discussed as well as the strength and weaknesses of the Myo as compared to other devices. The review will be finalised by the conclusion on the State of the Art and the preliminary answer to RQ1.

2.1 Myo armband specification

The Myo, which can be seen in figure 2, has been released by the Canadian company Thalmic Labs in 2013 and can currently be bought for $199 [1]. It is an armband which, when worn on the widest part of the forearm, is able to measure EMG (the amount of muscle activity) and motion and is able to interpret these data into gestures. Currently, there are five pre-programmed gestures provided by Thalmic Labs, which can be seen in figure 3. With enough knowledge the raw EMG data can be accessed in order to create one's own gestures via the Software Development Kit. The data is send over Bluetooth, which results in a range of at least 15 meters, and can be used by computers, tablets and smartphones running Windows, OS X, iOS or Android operating systems.

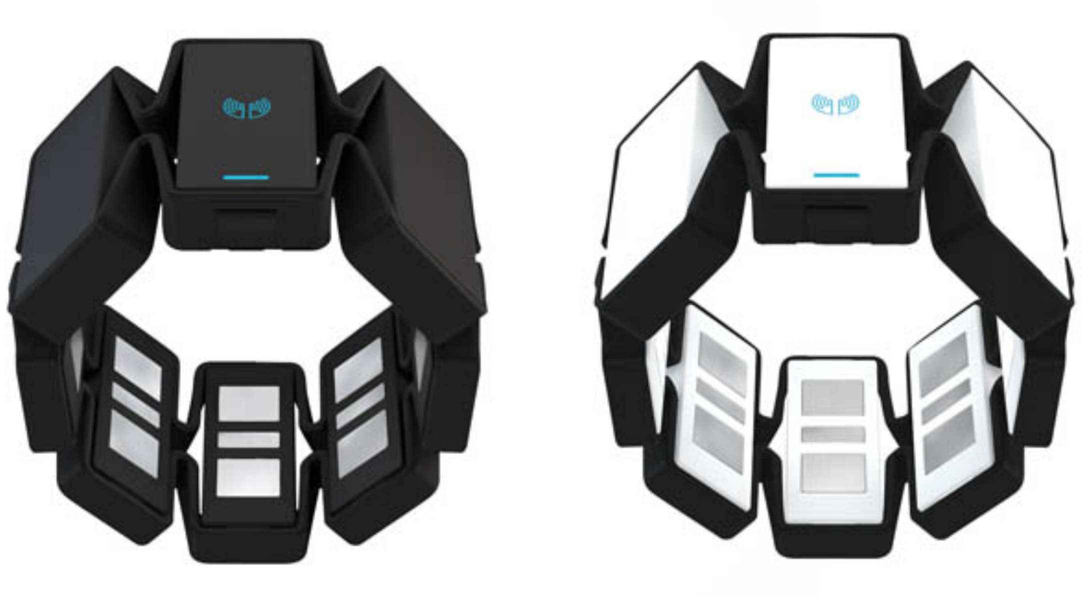


Figure 2: The Myo armband in the two possible colours [7]

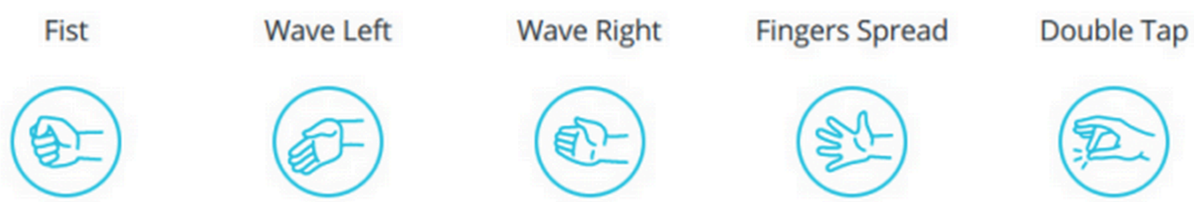


Figure 3: The five pre-programmed gestures of the Myo [8]

2.1.1 Hardware

The Myo is delivered in a package containing, besides the Myo itself, also a Micro-USB cable, a Bluetooth adapter and 10 sizing clips [9]. The Micro-USB cable is used to charge the Myo and can be used to turn on the Myo. The Bluetooth adapter is used to connect to a Windows or Mac computer. The sizing clips can be used to make the Myo fit more tightly around the arm by applying the clips onto the Myo band. The band is expandable from 19 to 34 cm circumference and the Myo itself weighs 93 grams.

The sensors on the Myo consist of eight medical grade stainless steel EMG sensors and a nine-axis Inertial Movement Unit (IMU), containing a three-axis gyroscope, a three-axis accelerometer and a three-axis magnetometer [9]. To aid the EMG sensors in providing the correct data a 50 and 60 Hz Notch filter is applied to filter out the power line interference [10]. The EMG signal also goes through a quad operational amplifier which filters the unwanted signal and amplifies the wanted signal [11]. The benefit of this configuration is that the surroundings have little influence on them.

The Myo has two ways of providing feedback. The first is through haptic feedback provided by a vibration motor. This vibration motor is able to provide short (23.5 ms), medium (45.6 ms) and long (100.5 ms) vibrations [12]. The vibration motor as well as other components located on the central pad can be seen in figure 4. The second way is through two LEDs located on the central pad of the Myo. These LEDs are used to indicate the battery, connection and firmware status [13].

The battery of the Myo consist of two 260 mAh lithium ion batteries, located at either side of the central pad [11]. These batteries should be able to last up to one full day on a single charge [9].

The communication with the Myo is done via Bluetooth. The transponder here of is also located on the central pad. This allows the Myo to connect to mobile phones with a built-in Bluetooth radio (which almost all phones have) and with the Bluetooth adapter once it is inserted into a Windows or Mac computer.

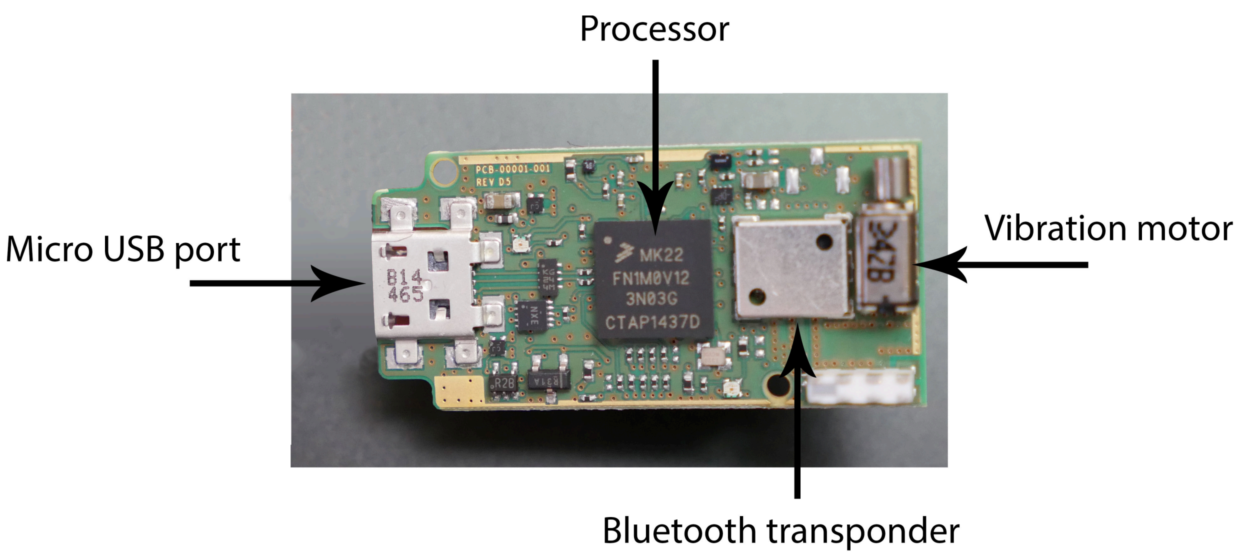


Figure 4: Central pad of the Myo [11]

2.1.2 Software

To connect with the Myo first the Myo Connect app by Thalmic Labs has to be installed [1]. The Myo Connect app handles the data sent from the Myo to the device one is using and allows the Myo to interact with other applications on the device. In this app the Myo can be calibrated, turned off, renamed and pinged, it is also possible to create a personal calibration profile. The calibration starts whenever the Myo is first connected, but using this app it can be redone whenever needed. This calibration only requires the user to wave outwards. However, the accuracy of the EMG is increased when the sensors have warmed up (takes about two minutes) at which point the user is asked to wave outwards again. Normally the Myo turns on whenever it is moved, this can cause the Myo to drain its battery even though it is not being used. After it has been turned off in this app, it can only be turned on again by connecting it with the Micro-USB cable. The personal calibration profile is useful if the user wants the Myo to be more accurate. This is done by performing each gesture once after which a personal profile is created. Since the muscles to move the various fingers are too close together, the Myo cannot be used to clearly read out the position of the fingers. However the accuracy to read out the pre-defined gestures provided by Thalmic Labs rarely are inaccurate.

A developer is able to create his own applications by using the Software Development Kit (SDK) provided by Thalmic Labs. When the application is finalised and the developer wants to share it, the application can be uploaded to the Myo Marketplace.

In figure 5 the general overview of the communication with the Myo can be seen. At the top is the application written by the developer, the application can be written in C++, C or Lua [14]. However the community has created other libraries to also be able to write in other programming languages. This application will then connect to the libmyo library which connects to the Myo itself via the Bluetooth connection.

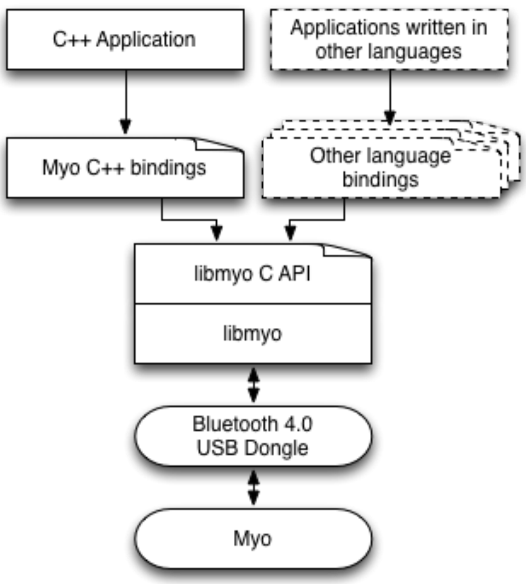


Figure 5: Application overview [15]

The earlier mentioned Myo Connect app is delivered with some applications built-in. These applications are mostly connectors to different programs, such as Youtube or Netflix. If wanted more applications can be downloaded from the Myo marketplace, where all the uploaded applications for the Myo are located. More on the available applications will be discussed in section 2.3.

2.2 Background research

To aid in the creation of an application for the Myo a literature review has been done regarding the creation of a gesture recognition system for the Myo. The goal of this review is to gain insight in how to successfully create a gesture recognition system for the Myo. Which will be used later in the creation of a unique application for the Myo as is the final goal of this graduation project. This review will focus on what steps have to be taken to recognise gestures with EMG and acceleration and how these steps can be implemented. Later, the review will also clarify on how to implement these steps onto the Myo specifically.

To accomplish this goal first the unique features of EMG and acceleration will be explained. Second, the general steps that have to be taken to use these features for recognition. There after, these steps are explained in a more extensive way, combined with a probable structure for one's own gesture recognition system. Finally some remarks on how these steps have to be implemented on the Myo and possible setbacks of the Myo will be made.

2.2.2 Gesture recognition methods

There are currently three main gesture recognition techniques. These techniques will be discussed more elaborately in section 2.3.2. First is gesture recognition based on computer-vision, such as the LEAP motion or Kinect. A downside of this technology is that it is sensitive to changes in the background. Another downside is that it requires the user to be in a confined space [16], [17].

The second technique is the dataglove, which requires the user to wear a glove that recognises the movement of the hand by using bending sensors. This, however, requires the user to wear a movement-limiting, inconvenient to hold glove [16].

The third technique is the one implemented by the Myo. This technique involves the combination of electromyography which measures the muscle activity of the arm and an Inertial Measurement Unit, which measures acceleration and orientation [16]. The acceleration data can be used to recognise large-scale gestures, such as the movement of the arm. While the EMG signals are used to capture the muscle activity related to finer motions such as wrist and finger movements [18].

2.2.3 Gesture recognition steps

There are four steps that have to be carried out in order to recognise a gesture using the Myo. These steps are the pre-processing of data, segmentation, feature extraction and classification [10], [16], [18]-[22]. These steps will be further elaborated upon below.

Before a program can recognise a gesture first a dataset has to be created [10], [21], [23]. This dataset contains the gestures that have to be recognised and has to be filled with training data. The amount of training data differs from 4 [20] to 32 per gesture [18] and is dependent on the classification method that has been used. This is not stated as one of the four steps mentioned earlier, since this only has to be done once, while the earlier mentioned steps have to be performed each time a gesture has been made.

Pre-processing

To make the data easier to work with, the data is pre-processed. The pre-processing of the acceleration and EMG is done separately. The final goal for the pre-processing of the EMG data is to obtain the power exerted by the muscle. The first step for this is to remove the DC offset of the signal, by applying a high-pass filter [16]. Further, by rectifying the signal, the negative values of the signal are removed [10], [16]. There after, it is important to remove the high-frequency noise. This can be done with either a Moving Average filter [10], [19], which also smoothes the signal or a low-pass filter, which just removes the high-frequency noise [16]. The signal can also be down sampled to increase the processing speed [10].

The pre-processing of the acceleration data also consists of various steps. The first step is to remove the high-frequency noise by either applying a low-pass filter [10], [20] or a Moving Average filter [16], [18], [22]. Next, the DC-offset or gravity can also be removed by applying a high-pass filter [20], [22]. Finally, the data can be made easier to handle by normalising the signal [10], [18]. Sometimes the data shows a constant movement towards a certain direction even though the device is stationary. This constant movement is called drift. Banierink [10] suggests to remove the drift by setting the acceleration at (near) stationary periods to zero. The probable drift is then calculated by the acceleration differences between the stationary periods. This drift can then be subtracted from the signal.

Segmentation

Segmentation of the data is done to acquire the active part of the signal. The active part of the signal is where the gesture is made [19]. There are two methods to determine in realtime where the gesture is made without the use of additional, external sensors.

The first method is best to use when the gestures all use either EMG or acceleration. Zhang et al [19] and Lu et al [18] do this by comparing the EMG signal with a set onset value. However, this can also be done with acceleration. Whenever the signal is higher than the onset value for a set amount of time the gesture is started. There after, whenever the signal is lower as the offset value (which is lower as the onset value) for a set amount of time the gesture is stopped.

The second method is best to use when the gestures are not all defined by EMG or acceleration. This method consists of scoring the data upon different variables. Xu et al [20] determine five different variables, the amplitude, point separation, mean value, distance from nearest intersection and sign variation. These variables are scored and whenever the score is above a threshold a terminal point is set. The other terminal point is set whenever the score is below the threshold. To acquire the complete gesture the terminal points are checked to see whether some are too close together, these are then joined together to create one gesture.

Other segmentation methods are also mentioned, however these are not done in realtime [16] or use additional, external sensors [10]. These methods are therefore not useful if one wants to create an application that has to work using solely the Myo.

Feature extraction

The goal of feature extraction is to select features from the data and extract this into a matrix which can be fed into the classifier for the final step. For the EMG data the moving average value should be extracted [10], [16], [18], [19]. Furthermore, authors argue that either an autoregressive function should be used [18], [19] or that the standard deviation [16] or the peak placement [10] should be extracted.

The authors disagree on what method should be used to extract data from the movement data. The most mentioned method is using the standard deviation [16], [19], [22] and mean [19], [22] as the classifiers. It is also possible to extract more types of data as argued by Wang et al [22], who use up to eight methods to extract data from the movement data, also using the variance, inter-quartile range, correlation, mean absolute deviation, root-mean-square value and the energy. Lu et al [18] suggest to use the down sampled acceleration variable, the variable representing the amount of arm movement and the variable representing the difference between start- and endpoint. The features to be extracted hugely depend on what classification method being used, therefore not a single best one can be selected.

Classification

Two classification methods came forward. The first method, a self-learning algorithm, is demonstrated by all authors [10], [16], [18]-[20], [22], [24]. The authors however do not agree on which self-learning algorithm to use. If the data is easily separable, the use of a decision tree is recommended [18], [19]. Such data can for instance be seperated between a gesture using only finger and hand movements or gestures which use the arm as well[18]. Whichever self-learning classifier would be best is unclear, since no author argues why one method was chosen over the other. The methods which are used by the authors are a Hidden Markov Model [19], [24], an Artificial Neural Network [10] [16], a Bayes linear classifier [18], a Hopfield Network [20] or a Probabilistic Neural Network [22].

The second method is suggested by Lu et al [18], which is the use of a scoring method to identify the best fit. During this method all the variables from the feature extraction are scored based upon how close they are to the training gesture. There after, the highest score is selected as the gesture. It is worth noting that the variables are compared to all the training data, therefore the amount of training data is limited to 4 samples per gesture to reduce computation time.

2.2.4 Implementation using the Myo

Although most authors did use the Myo for gesture recognition, no author mentioned practical steps for implementing in the Myo specifically. Some authors did mention some remarks that should be taken into account when implementing a gesture recognition system into the Myo. First, Banierink [10] mentions that because the Myo has eight EMG sensors the features also consist of eight channels when extracting from the Myo. Second, Vorapojpisut et al [24] state that the Myo supports some software languages. However, if needed, more libraries have been created by the community to also connect to other languages or programs. Vorapojpisut also recommends to use the Matlab program, because it offers quite some signal processing functions.

2.2.5 Conclusion

In the previous paragraphs the way to create one's own gesture recognition system is explained. This research was done to later implement into a gesture recognition system into the Myo.

Four different steps were analysed to recognise a gesture. The first is data pre-processing, which converts the data to be more easily usable in later steps. Second is segmentation, which recognises when a gesture is performed and divides the data in active segments. Two methods were analysed, the use of a threshold value and the use of a scoring method. Third is the feature extraction where specific features are taken out of the data which are put into a matrix to be used to classify the gesture. The final step is the classification of data, which implements a self-learning algorithm to classify which gesture has been performed. If needed a scoring method can also be used as a classifier. However this significantly reduces the maximum sample size due to comparing with every gesture.

First, since no authors mentioned the practical implementation or precise guidelines of gesture recognition into the Myo specifically this literature review is still lacking. It is recommended to do further research on the practical implementation into the Myo. Second the suggested structure as a combination of the literature has not been tested, therefore it might be possible that it would not be accurate or would not work at all. Third, since the authors tend to use different self-learning algorithms and none provided argumentation on why one was chosen over the other, no self-learning algorithm has been recommended. A further research into the different self-learning algorithms in combination with a gesture recognition system is therefore recommended.

2.3 State of the Art Research

Here the research into the applications of the Myo and into comparable systems will be discussed. First the used research method will be discussed. Next the research into the current Myo application domains will be discussed. There after, the research will continue with a discussion on comparable systems to the Myo armband.

2.3.1 Research method

The goal of the State of the Art research is to create an overview of comparable devices to the Myo to search for possible unused domains and to gain insight in the added benefit the Myo provides. A research into the applications already created for the Myo has also been done in order to divide them into different domains. After this is done it should be possible to select a suitable research domain for the rest of the project.

The research into the current applications will be done via Google Scholar, using keywords such as “Myo armband applications” and “Myo Thalmic Labs”. The Myo website will also be used as a source. First, the Myo marketplace offers all the publicly accessible application currently created for the Myo. Second the developerblog and press page offer a good insight into the research done with and into the Myo.

The research for finding comparable systems to the Myo, Google will be used. Since most of these systems should be consumer devices, they should be marketed online and therefore easily be found through the Google search engine.

2.3.1 Myo application domains

The research into the current Myo application domains is aimed towards answering the first subquestion: “What are the current applications the Myo is being used for?”.

As for the current Myo application domains a research into this has already been done by Rutgers [25]. Rutgers defined five categories in which all applications of the Myo can be divided. These categories are:

- Virtual Reality
- Entertainment
- Robotics
- Medical
- Mouse & Keyboard replacement

Since her research has taken place a year earlier, a research has been done to verify whether these categories are still valid. This research has been done according to the aforementioned method. This research offered no new insights into the above stated categories and therefore the categories found by Rutgers are still viable.

2.3.2 Comparable systems to the Myo armband

The research into comparable systems is done to look for possible application domains which currently do not exist and to gain insight into the added benefits of the Myo. The result of this research would provide the answer for the second and third subquestion: “What are possible applications for the Myo which currently do not exist?” and “What is the added benefit the Myo provides?”.

The Myo armband is quite unique when it comes to using EMG and motion data to recognise gestures. However there are other products which also recognise gestures, for instance the LEAP motion or Kinect. There are also other products which measure bio signals, such as heart rate or arm movement, at the forearm, for instance the Fitbit and LG G watch.

Gesture recognition systems

The main feature of the Myo is that it is able to recognise different gestures. Although the way the Myo does this is quite novel, it is not the only gesture recognition system. Currently there are a couple of other gesture recognition products on the market, which will be described below. In the conclusion a summary of the strengths and weaknesses of the different gesture recognition devices can be found.

LEAP Motion [26]: The LEAP Motion is a small controller which can be linked via USB to a computer. The device recognises the hands moving above the device and allow the user to control the computer with movements and gestures handsfree in 3D space. The LEAP Motion does this with two cameras and three infrared LEDs. The cameras pick up the reflected infrared light and the LEAP Motion converts these images to useable position data using clever algorithms. The LEAP Motion can be seen in figure 6.

The LEAP Motion offers two main implementations. The first is the use in combination with a virtual reality device, such as an Oculus Rift. The LEAP Motion is then located on the device allowing it to read out the hand gestures made in front of the user allowing for a greater immersion.

The second main implementation is intuitive, instant, handsfree interaction with other devices. Since the device does not require any calibration, anyone moving their hand in front of the device can directly use it. Also because of good user interaction design most of the applications are easy to use. An example of good user interaction is swiping in mid air to move to the next slide or using an open hand to rotate the view around.

Figure 6: The LEAP Motion connected to a computer [27]

The LEAP Motion offers a sub-millimeter accuracy read-out of the movement of the hands and fingers. However if the vision of the hand is lost the device cannot track the hand anymore. This can happen if the hands are moving over each other or when the hands are moved out of the 25-600 mm vision range. The device is also sensitive to environmental intrusion, such as bright lights, which can prevent the device from registering the hands. The LEAP Motion can be used directly without calibration or other preparation. It does not have a battery and thus should always be connected to a computer. During usage the LEAP Motion offers a handsfree, non-intrusive interaction. The LEAP Motion costs $79,99 in total.

Kinect [28]: The Kinect has originally been a game controller developed by Microsoft. However, since its unique interaction and the availability of an SDK, it currently is used for much more as just gaming. The Kinect recognises full body movements. It does this by first creating a depth map by analysing a speckle pattern of infrared laser light after which an algorithm determines the location and pose of the body. Furthermore, it has the ability to locate where sound is coming from using a microphone array. The Kinect can be seen in figure 7.

The Kinect offers two main implementations. These are relatively the same as compared to the LEAP Motion with the first being augmented and virtual reality. An example for this is a clothes fitting application [29] which displays the user on a screen and places different clothes on the user to see how they would look.

Figure 7: The Kinect by Microsoft [30]

The second implementation is also handsfree interaction. However since the Kinect was released with games in mind most of these interactions are also aimed towards games, whereas the LEAP Motion is more targeted towards the controlling of devices.

The Kinect is aimed at recognising body movements instead of the more precise hand movements. It does this with a minimum accuracy of 4 centimeter at the maximum range of 5 meter, while at the minimum range of 0.5 meter the accuracy is only a few millimeters. The Kinect can read out the movement of the arms. However for the precise finger movements the device is not accurate enough, the Kinect also has trouble dealing with environmental intrusions, such as a busy background or too much light. The Kinect can be used without any calibration and also during usage it provides a handsfree interaction. The Kinect also has no battery and should therefore always be plugged in. The Kinect costs $109,99.

Dataglove: Another example of a gesture recognition system is a dataglove, such as designed by Paredes-Madrid [31]. Different from the before mentioned systems the dataglove is a concept and not an actual brand. The concept does have some start-up consumer devices, such as Manus VR [32] (see figure 8). However the prices are still high and there are very few applications. The idea behind a dataglove is to use force sensors to measure the force applied by the fingers to track their movement.

The implementations of datagloves are usually to control robots, such as a robotic arm. Another implementation of a dataglove is the combination with virtual reality to create for instance a surgical simulator [33].

The dataglove by Manus VR uses wireless bluetooth connection allowing for a range of at least 15 meters. The movement of the fingers can be measured very accurately with an accuracy of approximately 3 degrees per finger joint, which results in an accuracy of about 1 millimeter. The positional tracking is done via an IMU which results in a less accurate measurement as the image-based systems such as the LEAP Motion. This is since a small inaccuracy in the acceleration measurement can result in a larger position error. The IMU of the dataglove does require to be calibrated before each use, which usually takes less than 5 minutes. Also the set-up requires one to put on the gloves themselves, which also increases the intrusiveness of the device. Since the Manus VR does not rely on images it is not as intolerant to changes in the environment as above mentioned devices. The battery lasts between 3-6 hours. The final price tag for the Manus VR is

$1000, including the full development kit. If the Manus VR comes out of its development phase it probably will become cheaper.

Figure 8: Manus VR system [34]

Forearm-wearable sensor systems

Aside from the gesture recognition the Myo has another aspect, which is that it is wearable on the forearm. Again this is not a unique feature, since there are more sensors which are worn on the forearm. Below the differences with these kind of products will be discussed.

Fitbit [35]: The Fitbit is an armband which is worn on the wrist. It is able to recognise the amount of steps taken and calculate the amount of calories burned. There are also versions, such as the Fitbit Alta HR which can measure the heart rate. The Fitbit Alta HR can be seen in figure 9.

The Fitbit aids people in living healthier by promoting exercise, but also by giving an overview of their day. It comes with a mobile phone app, which can give an overview of one's exercise, one's sleeping rhythm and more. The Fitbit also functions as a smartwatch by showing incoming messages, by showing who is calling and time.

Figure 9: The Fitbit Alta HR [36]

Smartwatch: Currently, smartwatches are becoming more and more common. The functionalities of the smartwatches therefore also increase. The Apple Watch (see figure 10), for instance, features a heart rate sensor [37] and the smartwatches running Android wear can also be navigated with wrist gestures [38].

The main feature of a smartwatch is to display information on the wrist of the user, so the user does not have to take out their smartphone. It can also be used for simple interactions such as replying by voice recognition.

Figure 10: The Apple Watch 2 [39]

2.4 Conclusion

The State of the Art research was focused on answering the first research question: “In what domain(s) can the Myo provide actual benefit?”. To answer this question three subquestions had been formulated: “What are the current applications the Myo is being used for?” and “What are possible applications for the Myo which currently do not exist?”. The first subquestion has been answered by looking into the current applications of the Myo, these are proven domains in which the Myo can be used. The second subquestion has been answered by looking into comparable products to the Myo and looking into what fields they are used in.

When combining these researches, most fields mentioned by Rutgers [25] would remain the same. The only thing that could be changed is medical into health. The reason for this change is that, when looking into the Fitbit, the Myo probably also can provide benefit into the field of sports or healthy lifestyle, not only in the medical domain. This change is not to exclude the medical field from the research domains, it is to broaden the category.

Applying above change to the five categories as defined by Rutgers [25] would result into the following categories:

- Virtual Reality
- Entertainment
- Robotics
- Health
- Mouse & Keyboard replacement

The answer to the third subquestion: “What is the added benefit the Myo provides?” is related to what the Myo does better as its competition, which in this case are other gesture recognition devices. In table 1 a comparison between the various devices can be seen. This table clearly shows that the Myo should not be used if the user is looking for an accurate system. However the Myo does score well when looking at the range, power and intrusiveness categories. In the latter category the Myo does not score full points since the user has to wear the armband. However, the

Myo does offer a handsfree experience and thus barely limits its user. Furthermore, the Myo is the only device on this list which can measure the muscle activity. An application for the Myo should therefore focus on the importance of a large range and battery life as well as the usage of muscle activity, while also being able to use the hands for other tasks.

Function	Myo	LEAP Motion	Kinect	Dataglove
Range	+ +	-	+	+ +
Accuracy of hand measurements	-	+ +	- -	+ +
Accuracy of arm measurements	+	+ +	+ +	+
Calibration/ preparation time	-	+ +	+ +	-
Power	1 day	No battery	No battery	3-6 hours
Intrusiveness	+	+ +	+ +	-
Sensors	EMG and acceleration	Visual	Visual	Force and acceleration
Price	$199	$79,99	$109,99	$1000

Table 1: Comparison between the various gesture recognition devices

3. Methods and techniques

An overview of the different methods and techniques that were used during this project will be provided in this chapter. First, the Creative Technology Design Process is discussed, after which the various methods which come forth when discussing this process will be elaborated upon.

3.1 Creative Technology Design Process

The Creative Technology Design Process as designed by Mader et al [40] consists of four phases: Ideation, Specification, Realisation and Evaluation phase. The first three phases are designed as such that these phases starts by diverging, widening and exploring all possibilities, after which the converging starts where the best idea, specification or product is selected. An overview of the design process can be seen in figure 11.

3.1.1 Ideation phase

The goal of the ideation phase is to go from a user need, technology or creative inspiration to an actual product idea. In this project brainstorming will be used to gather ideas as well as looking at related work, which has already been done in the state of the art. The expert review as well as own judgement are used to finally select a product idea. To further clarify the idea a stakeholder analysis as well as use scenarios will be created.

3.1.2 Specification phase

During the specification phase the product idea will be elaborated upon to reach the product specification. During this phase a requirement analysis, functional analysis and a system architecture will be developed.

3.1.3 Realisation phase

The realisation phase will result in an actual prototype created according to the product specifications of the previous steps. This is done by first creating the various components of the prototype and then slowly working towards the final prototype by combining the components. During the realisation phase the functional evaluation should already take place by validating whether the components meet their specified requirements and function as defined by the product specification.

3.1.4 Evaluation phase

The evaluation phase consists of two parts, the first being the final functional evaluation. Although the prototype should be checked during the realisation phase, one final functional evaluation is done to see whether the final prototype adheres to all the specified requirements and functions. The second part is the user evaluation. During the user evaluation the user experience is tested and the earlier stated requirements are verified whether they result in the intended interaction.

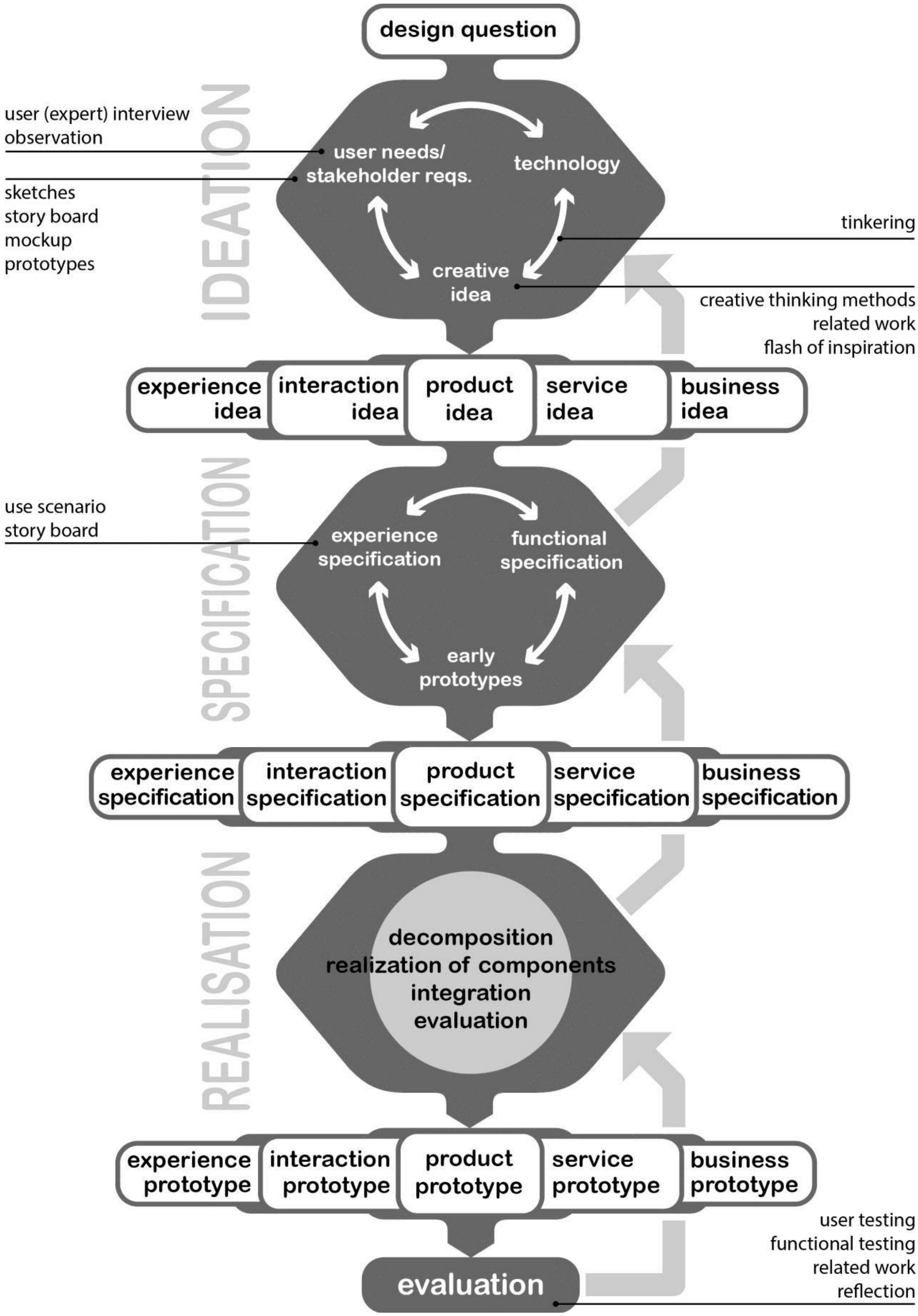


Figure 11: The Creative Technology Design Process

3.2 Brainstorm

The brainstorm for new ideas will employ two techniques, free-form brainstorming and mind mapping [41]. For both techniques four main rules are important [42]. First, criticism is ruled out during ideation. Second, unconventional ideas are welcomed. Often the final idea is based upon an unconventional idea. Third, quantity is wanted. The more ideas generated, the greater the chance a successful solution will be found. And fourth, the combination of and improvement on ideas are wanted.

3.2.1 Free-form brainstorming

Free-form brainstorming is done with a group. First, the problem or assignment for which a solution needs to be sought is given. There after, everyone is able to give their ideas. These ideas get written down visible to all participants so that ideas can be based upon each other. The benefit of this type of brainstorming over other group-based techniques is that this is easy to set up and can be done relatively quickly.

3.2.2 Mind mapping

Mind mapping will be done individually within this project. A mindmap starts by writing and encircling the main concept in the middle of a sheet of paper. From this concept various branches are drawn which describe a component of the concept. These components can also be split in branches of various subcomponents. This practice continues until no new branches can be drawn or new branches do not contribute to new ideas.

3.3 Expert review

During the expert review an expert in product design, in this case the project supervisor, will look into the various product ideas. The expertise of the expert will then be used to evaluate the various ideas. The result will be a collection of viable product ideas of which one will be chosen to actually create a prototype for.

3.4 Interviews

Interviews are a method for acquiring qualitative data. An interview consists of an interviewer, who is the person asking the questions and an interviewee, who is the person answering the questions and providing information to the interviewer. There are various methods which describe how the interview should be performed. Crabtree et al [43] describe three methods for conducting an interview:

Unstructured: Although no interview is truly unstructured, this method describes the interviews which are equivalent to guided everyday conversation. This type of interview is usually done in combination with a participant observation.

Semi-structured: When conducting a semi-structured interview a series of open-ended questions will be prepared by the interviewer. Other questions will emerge from the answers to these open-ended questions. The semi-structured interview can also be done in a group. However, this is not a shortcut to quickly doing multiple individual interviews at once, since someone will answer in another way when interviewed separately [44]. This type of interview will be used when interviewing in the ideation phase.

Structured: The structured interview mostly resembles a spoken questionnaire. This is because all the questions have been prepared beforehand and no other questions are allowed. It is best to only do a structured interview when sufficient trustworthy information has been collected beforehand.

Structured interviews usually lead to a set of quantitive data. This type of interview will be used for the survey in the evaluation phase.

3.5 Stakeholder analysis

‘A stakeholder in an organisation is (by definition) any group or individual who can affect or is affected by the achievement of the organisation’s objectives.’ [45]

A successful stakeholder analysis starts by brainstorming all the possible groups/people which might affect or be affected by the product [46]. All possible stakeholders should be written down, the stakeholders which are important will be defined later. According to Sharp et al [47] there are four types of stakeholders: Those who will interact with the system (Users). Those responsible for creating the system (Developers). Those who provide guidelines for operation that will affect the development and/or operation of the system (Legislators). And finally, those in decision-making structures that relate to the system under development (Decision-makers). The list of stakeholders should be divided into these stakeholder groups.

The next step is to create a power-interest matrix, which maps the various stakeholders into a matrix based upon the power and interest they have in the project (both according to a 1-10 score) [46]. The matrix can be seen in figure 12. The matrix shows with which stakeholders to interact most (Players) and which to give minimal effort to (crowd). The subjects should be kept satisfied and the context setters should be kept informed [48]. It is important to keep in mind that the power-interest matrix can change at any moment and should therefore be kept up-to-date.

Finally the motivations of the stakeholders should be analysed and whether the stakeholders are proponents or opponents of the project [48]. The final analysis should be used for example in determining who to do user tests with and who to involve in the definition of requirements.

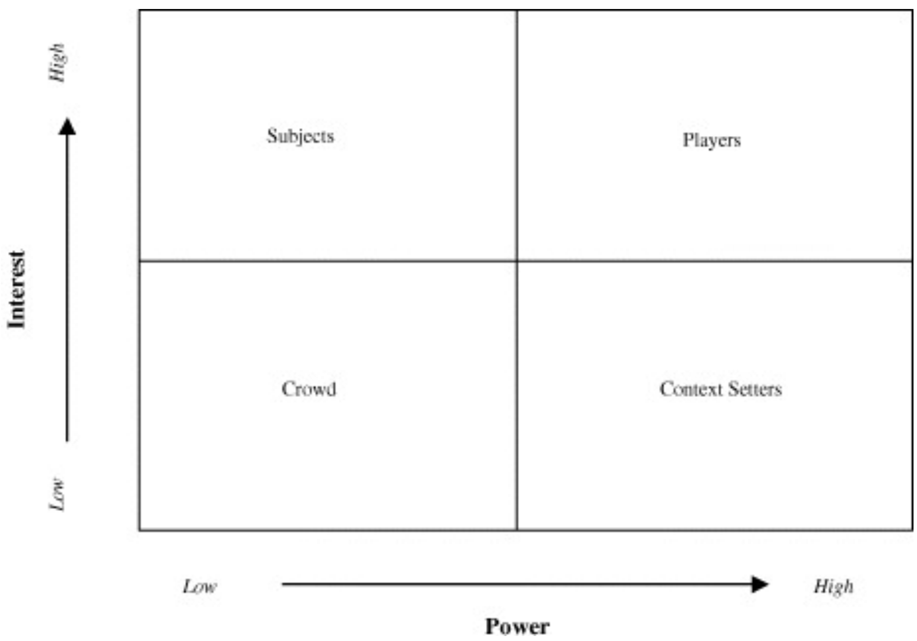


Figure 12: The power-interest matrix of stakeholders [46]

3.6 Personas

A persona is an archetype of a user that is given a name and a face, and it is carefully described in terms of needs, goals and tasks [49]. When designing for users the name of this persona is used rather than saying “the user”. This is because “the user” is not specific enough and might be subject to change, while when designing for a specific persona all designing is done with the same goal in mind. Various personas can be created, however every persona should have a different goal, otherwise the personas can be merged together.

3.7 iPACT

To gain insight into who will use the system and how they will interact with the system an iPACT analysis will be made. The iPACT analysis will describe the intention, people, activities, context and technologies related to the product [50]. The intention is aimed at describing the goal of the system, what it is aimed to do. People will describe who will use the system, which is usually done in the form of a persona (a fictional user). The activities are a description of various situations in which the product will be used. The context is used to describe the environment or location where the product will be used. The technology describes what essential technologies will be used by the product. This analysis can further be used to create a use scenario. This is a story about how the system will be used, showing the interaction with system.

3.8 Requirement analysis

Requirements analysis is a structured method of describing what the system should encompass and how it is supposed to work [51]. This is described in a list of separate requirements which each define a single part of the system. The requirements will be categorised using the MoSCoW method and divided into functional vs non-functional requirements. These methods will be further elaborated on below.

3.8.1 MoSCoW

The MoSCoW method is used to represent four different priorities of requirements [52]. The four different priorities are Must have, Should have, Could have and Won't have. Must have requirements are requirements which must be met in order to not fail the project. Should have requirements define requirements that would be nice to have. The same can be said for Could have, however these are less important as the aforementioned. Won't haves describe requirements which will not be implemented in the current project. They may be requirements which could be done in a later stage of the project.

3.8.2 Functional vs non-functional requirements

A commonly used distinction between requirements is to separate them into functional requirements and non-functional requirements. Functional requirements focus on the functions a system must be able to perform [53]. There are however extensive discussions about what non-functional requirements actually entail according to Glinz [53]. The definition used here comes from Robertson and Robertson [54]: A property, or quality, that the product must have, such as an appearance, or a speed or accuracy property.

3.9 FICS

The FICS is a way to describe the system from the designer-perspective [55]. The FICS and iPACT are closely related. The iPACT describes the system from a user-perspective focusing on intended usage and feel of the system, whereas the FICS describes how the system works and what happens with the input provided by the user. FICS stands for Functions and events, Interaction and usability issues, Content and structure and Style and aesthetics. Functions and events describe what the system does and how it reacts to certain events. Interaction and usability issues describe how the user is intended to use the system. Content and structure is the backbone of the system, this is how the data is stored and how to access it. Style and aesthetics describe the look and feel of the system.

3.10 System architecture

A system architecture is a template for a family of systems [56]. The goal of the system architecture is to show in a technical view how the system works. The system architecture shows various components and how these are connected to one another. As an analogy, architectures can be viewed as printed circuit boards where the various chips can be viewed as components and the wires in between show how the various components are connected to one another [56]. Usually a system architecture can consist of various levels, where each level dives deeper into a separate component, revealing the inner workings of this component as a set of sub-components.

3.11 Evaluation

In this graduation project two types of evaluations are performed. The first being functional testing, which is used to test whether the application is actually working and second the user evaluation which tests what the users think of the application and to test whether the requirements are actually met.

3.11.1 Functional testing

Functional testing involves testing by the developer whether various features of the application work as intended. The functional testing should be performed before the user evaluation is started, since this prevents or limits the changes of errors showing up during the user evaluation. The goal of these tests is to first determine whether the tested feature works and if not what the error is. Later on these errors can be fixed. Another functionality test should be done afterwards to see whether the feature does work now.

3.11.2 User evaluation

The final application created in this graduation project will be evaluated via user evaluations. For the setup of the evaluation the setup provided by Ellis et al [57] will be used. Here it is important to first think of the purpose of the test. This purpose will be to see whether the set requirements are met and accepted by the users as well as seeing what still can be improved in the application. It is also important to both acquire qualitative and quantitative data. This will not only allow one to see how often something occurs, but also uncover the reason behind it. Furthermore, it is important to ask whether what is being asked is related to what one wants to measure (for instance, likability should not be asked by asking for usability).

To uncover these experiences a fixed assignment should be created, which allow for the measurement of variables across different users, since it keeps the data consistent. Furthermore, the questions asked to the users should also be consistent. This can be done best with the usage of a survey or structured interview. Within this survey a Likert scale can be used to acquire quantitive data, while open-ended questions can be asked to acquire the qualitative data.

4. Ideation phase

The goal of the ideation phase is to come up with various ideas for applications of the Myo, select a certain idea to continue with and clarify who will use this application as well as how it should be used. The methods for the execution hereof have been explained earlier in chapter 3.

4.1 Idea generation

To generate as many ideas as needed a brainstorm is performed. The basis for this brainstorm was from the technology aspect of the product, the Myo. First, an individual brainstorm was done by creating a mindmap. After which a group brainstorm has been performed. Later on a brainstorm based upon a small subsection of categories was performed. The best ideas were then selected in meetings with the supervisor as expert review as well as an own selection based upon personal interest and uniqueness.

4.1.1 Individual brainstorm

The individual brainstorm was done by creating a mindmap, which started with the five state of the art categories as setup. Now ideas and subcategories were added unto the mindmap. When the brainstorm came to a halt different categories or ideas were combined to look for new possibilities. The end result of the mindmap brainstorm can be seen in appendix A.

4.1.2 Group brainstorm

The group brainstorm was performed with 5 people. The brainstorm started by showing and explaining the Myo, after which a first set of ideas was created. These ideas were written down on post-its creating a physical mindmap. When the idea generation came to a halt the five state of the art categories were introduced as an aid to continue the brainstorm. After about 15 minutes no new ideas were generated and thus the brainstorm was ended. The ideas were then put into a mindmap, which can be seen in appendix B.

4.1.3 Category brainstorm

After the end result of the brainstorm was discussed with the supervisor various subcategories were found. These subcategories are add-ons, serious games, environment interaction, education and communication. It was also suggested to think of the question “Can I use the Myo for this?”, during everyday tasks. A brainstorm was then performed to think of ideas using these subcategories and the question. The result of this can be seen in appendix C.

4.1.4 Expert review

To aid in the converging of the ideation phase an expert review has been performed. The expert review was done by consulting with the supervisor. First a combination of the first two mindmaps was discussed, after which an agreement was made to perform the third brainstorm. The result of this brainstorm was also discussed and together with own converging resulted in three application ideas which would undergo a feasibility research. These three ideas which will be tested are:

AirBoard: Currently teachers are using various tools to communicate information, such as slides, whiteboard or SmartScreen. Each of these tools have their benefits as well as downsides. AirBoard will bring forth another tool which can be used by teachers which allow the teacher to write in thin air. This will then be displayed on a display in front of the class. This allows the teacher to talk towards the students while explaining or even use the board while explaining and standing next to a student.

Guitar trainer: This application will transform the Myo together with one's smartphone into a guitar allowing one to learn and play guitar anywhere. Using the Myo the user is able to hit the strings while the display of the phone allows the user to indicate which chord should be played.

Tennis trainer: This application will give insight to the user on how he/she is performing in tennis. The user will wear the Myo while playing a game of tennis. There after, an analysis of the strikes of the user will be made which is compared to a database containing how various strikes should be done. The application will then present the user with an overview of how these strikes could be improved by for instance stating that a strike should contain a more subtle wrist movement.

4.2 Feasibility

The selected ideas will be tested on feasibility in two ways. The first method will be a check with the conclusion of the state of the art. In here the benefit of the Myo is explained and a possible application should tap into this added benefit. Second is by interviewing potential users or experts. These interviews will show the potential of the application as well as possible risks which might come up when using the application or in the creation of the application.

4.2.1 State of the art

In the state of the art of chapter 2 it was mentioned that the added benefit of the Myo mostly lies in the range, power and non-intrusiveness of the device. The unique nature of reading out the muscle activity instead of the movement of the fingers was also stated. The above mentioned ideas should therefore implement at least some of these benefits.

The AirBoard makes use of the range, power and non-intrusiveness of the device. Due to the Myo's range and power it can be used all day throughout the whole classroom. The teacher is still able to do everything he needs to do during his lessons, for instance a distillation experiment during a chemistry class. However, the AirBoard does not necessarily need to read out the muscles to function. The unique nature of reading out the muscles is therefore unused.

The Guitar trainer does not implement any of the above mentioned benefits. The only functionality the Myo is used for is to indicate when and in what speed the strings are struck. The same functionality could also be created using the LEAP Motion. For this reason the Guitar trainer will not be evaluated via an interview.

The Tennis trainer makes use of all the benefits of the Myo. Due to its range, power and non-intrusiveness the user can play a game of tennis without much hindrance. It is anticipated that the Myo is about as much hindrance as wearing a sweatband. Also the unique feature of reading out the muscle activity is explicitly used as input for the analysis of the grip of the player, which can not be achieved using other methods.

4.2.2 Interviews

As stated before, the Guitar trainer will not be further evaluated in interviews. The conducted interviews were done with a semi-structured set-up (see chapter 3.4). This set-up allowed for getting as much input as possible and allowing for the best way to gather information. Below the culmination of interviews per idea are discussed.

Airboard: For the feasibility test of the AirBoard three people were interviewed. Two were fellow students who have a side-job as teachers. Since they are relatively young they could be more eager to implement new technology. The third was a teacher, who is also working on introducing technology into education.

Interviewees foresaw difficulties for this application. The main difficulty lies in the attention span of the students. If the application does not work perfectly all the time, errors will occur with the potential risk that the teacher loses the attention of the students. Since the gesture recognition of the Myo is not perfect, errors will occur. This can even be seen in one of the main applications for the Myo, the presentation mode. Therefore there is a high chance this will happen when creating this application.

The interviewees did see the added benefit of the application. The most stated added benefits were being able to write from anywhere in the room. Another added benefit which came forward is that the teacher does not stand in front of what he is writing. The new technology might also help to gain the attention of the students, since they are interested in seeing something new happen.

Besides the earlier mentioned difficulty a lot of other negatives were also mentioned. First, the lack of feedback towards the teacher. It is harder to write correctly if one does not directly see what one is writing. It might be possible to lessen this if the interaction is well thought out. However, it will never be the same as a crayon or pen. Second, the difference in the needed writing technique was mentioned. Usually someone writes by moving the pen with the wrist, however the Myo is not accurate enough to measure the movement of the wrist. The Myo can recognise the movement of the entire arm. However, this is less natural and costs more energy. Third, the unintended use was also mentioned. Perhaps the Myo turns on on accident, resulting in a disturbance for the class. Last, the slow integration of technology in the classroom was stated. Although a lot of schools already have SmartScreens most teachers are not using them or using them correctly, since they are more used to the traditional methods and do not have time to invest in learning new techniques.

The question was raised how often a teacher would want to write something down when not standing in front of the board. The answer to this might be related to the chicken and the egg problem. When teachers are able to write from anywhere, they might want to do it more often.

The interviewees did state that they might be interested in using the Myo as remote to control the classroom, such as being able to black out all the computers for an explanation. Another interviewee mentioned that he would be most interested in the use of the Myo as a presentation tool.

Concluding, the interviewees were mostly enthusiastic about the idea. However, they were of the opinion that the required precision and accuracy requirements probably cannot be met. The interviewees also were concerned about how often the technology would be required.

Tennis trainer: The feasibility test for the tennis trainer was performed with four fellow students with at least some years of experience in tennis.

This idea was well received by the interviewees. A lot of input was offered regarding the types of hits and how these could be measured as well as what is important in tennis. It was stated that the footwork and the angle of the racket are two important parts of tennis, both of which cannot be measured by the Myo. However, it was also stated that it was important to grab the racket loosely, only tightening when the ball hits the racket. This can be measured by the Myo. Also the importance of fully finishing one's swing was mentioned and that a good swing should have a constant acceleration, reaching its peak of velocity when hitting the ball. The interviewees agreed that a good tennis swing is mostly performed from the shoulder, not the wrist. Both of which can be measured by the Myo.

Some challenges when building an application also came forward. First, all hits are different which significantly increases the difficulty of recognising the different swing types. It was also stated that sometimes the tennisser has to use a wrong technique just to get the ball over the net, which might lead to frustration if the application is giving negative feedback on these hits. The interviewees said that they would rather have a per hit feedback instead of a general analysis at the end of the game.

The speed at which the application can analyse will be important, a quick analysis can allow the tennisser to get feedback on a per hit basis instead of only giving a general analysis at the end of the game or training. The type of feedback is important to keep in mind. The interviewees differed in whether they would like a per hit analysis or a general overview. The per hit analysis could result in practicing “dry” swings (without a ball) to perfect the motion. The general overview allows the user to play their match without really paying any attention to the application yet still receiving feedback at the end.

The interviewees also offered some suggestions and raised some questions regarding the input for the application. Some mentioned that it could be helpful to create an application which requires two Myos. This would help in the practice of the serve, since it is important how the ball is thrown, which is done with the other arm. A second Myo could also help in detecting the turning of the racket, which would increase the amount of feedback that can be given. Another option for this would be to add a sensor to the racket which could detect the rotation. This would also help in detecting wrist movements.

The way feedback was provided was discussed as well. The idea of a visualisation of how the tennisser moved versus how he should have moved was received very well. A textual feedback was also well received, however this could increase the risk of the tennisser not understanding what he did wrong. It was also mentioned that feedback could also be spoken so the game can continue without having to run back and forth to the computer every time.

Finally the question was raised whether this application would replace an actual trainer. Some interviewees were against this idea and suggested that the trainer could use the provided feedback as input for its own feedback. One interviewee stated that he would use the application as replacement for a trainer, since he does not tennis regularly and therefore does not have a trainer. This application would then offer him a method of still acquiring feedback.

Concluding, the idea was well received and the tennis trainer application should be able to be created.

4.2.3 Final idea

For the choice of the final idea the above mentioned interviews will be taken into account, the amount as to which the application makes use of the added benefits of the Myo and how likely the idea will result in a working prototype.

Since the Guitar trainer did not use any of the added benefits of the Myo, this idea was rejected even before the interviews were conducted. As for the AirBoard, even though the application would make use of the freedom offered by the Myo, the interviewees expressed a concern as to the usefulness and accuracy of the application. The AirBoard would, however, result in a quickly useable paper prototype, since this would be an alteration of the already existing mouse application. The Tennis trainer was met with a lot of enthusiasm and the application would make use of the added benefit of freedom as well as making use of the readout of muscle activity to create an application which cannot be made using another method. The creation of a first working prototype however might prove to be more difficult, since no standard application which can be altered still exists. This would mean that it is also harder to involve potential users as early as possible in the design process.

Taking note of all the argumentations mentioned above the Tennis trainer is chosen to create an application for.

4.3 Stakeholder analysis

There are a couple of stakeholders related to the Tennis trainer. These stakeholders are identified using the method described in section 3.5. An overview of the analysis can be seen in figure 13 and the power/interest matrix can be seen in figure 14.

Users: There are two types of users, the first being the one wearing the Myo, namely the tennisser. The second being the optional coach or trainer which will use the application to gain further insight into the swings of the tennisser. The tennisser can also be divided into different users. First, the professionals, who want to perfect their swing and might also provide input for how the swing should be done. Second, the amateurs, these are tennissers who do train regularly with a coach and would use the application to get extra insight to further improve their game. Third, the hobbyists, these tennissers only play every so often, they do not train and thus do not have a trainer. For these tennissers the application could provide a good replacement for the trainer, giving insight and be available whenever the tennisser decided to play a game. the interest of both the trainer and the tennissers is high, since they will be the active users of the application. The power of the trainers is slightly higher as the tennissers, since the trainers better know what kind of insight is helpful for the tennissers and therefore will be taken into account more in the development.

Developers: There is only a single developer involved in this project, which is the developer of the application. Since the developer is the only developer in the project, its interest and power are both really high.

Legislators: There are two legislators in play: Thalmic Labs and the KNLTB (Dutch tennis association). Thalmic Labs is the creator of the Myo and can therefore decide against the application and therefore has a high power. The KNLTB might be interested in technologies that proves beneficial towards the sport. Both Thalmic Labs and the KNLTB have low interest due to the scale of this project.

Decision-makers: Two decision-makers have been identified: The developer and the supervisor. The developer decided how the application will look and therefore has a high power as well as a high interest. The supervisor has a relatively high interest in the project and power, since they are also interested in the correct finalisation of the project. The supervisor also sets the time frame of the project.

Stakeholder	Type	Power	Interest	Pro/con	Motivation
Tennissers - Pros	User	7	8	Pro	Perfect their swing Gain extra insight
Tennissers - Amateurs	User	7	8	Pro	Improve their swing Gain extra insight
Tennissers - Hobbyists	User	7	8	Pro	Easy access to a trainer
Tennis trainers	User	8	8	Pro/con	Gain extra insight Replacement of trainer
Developer	Developer / Decision-maker	10	10	Pro	A working and well-received product

Stakeholder	Type	Power	Interest	Pro/con	Motivation
Supervisor	Decision-maker	9	9	Pro	Good process and on time
Thalmic Labs	Legislator	7	4	Pro	New application
KNLTB	Legislator	4	2	?	Change within the tennis world

Figure 13: Overview of the stakeholder analysis

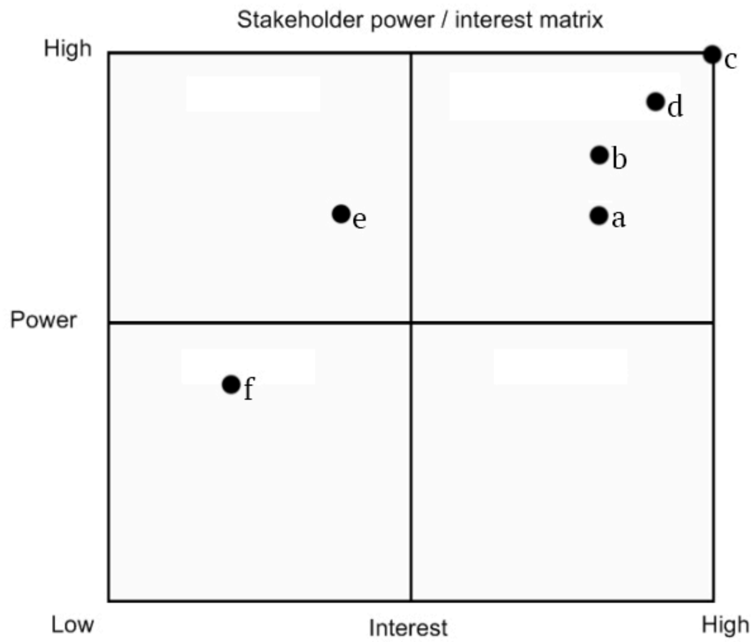


Figure 14: Stakeholder power / interest matrix. a) Tennissers b) Tennis trainers c) Developer d) Supervisor e) Thalmic labs f) KNLTB

4.4 Interview of expert client

To get a better overview of what is expected of the system an interview was conducted with a tennis trainer. Together with the responses of the tennis players interviewed in section 4.2.2 these interviews will be used in the creation of the preliminary requirements. It was intended to have multiple interviews. However, due to the lack of respondents of tennis trainers only a single interview has been conducted. This interview was done with a tennis trainer, because a trainer has a better insight into what kind of information is useful to a tennisser and what are actions which usually go wrong. The interview was performed in Dutch, the full untranslated interview can be seen in Appendix D. The most important parts will be discussed below.

The tennis trainer already uses video footage to give feedback to his trainees. He notices that this method already provides a lot of insight and helps the trainee to better understand what could be changed or improved.

The trainer was mostly interested in the power used by the tennisser during a specific movement. When is it high, when is it low. The power should be low at the beginning, when the racket is falling down to gain speed. Then when one is about to hit the ball the grip should tighten, after which the grip should loosen again and the swing should be fully finished.

The trainer noted that the main selling point of the system would be as an addition to the video feedback. This would mean that a video is shown with the current muscle power at that moment shown. This power could be shown in conjunction with the movement (as a 3D reconstruction), however a video footage should always be present. The main reasons for this was that it is easier for a user to understand the feedback if they can see themselves and it is more important for the user to have the correct movement rather than applying pressure at the correct time (someone can use the correct force for the swing, but if the swing itself is wrong it does not matter). Reconstructing the movement from the sensor data would probably not be correct in the exact movement and position of the swing.

The possibility of giving feedback to the user by the application based upon their movement and power would seem like a good idea. However, this requires someone with a perfect technique to perform the training data, so that the system gives the correct feedback. Also the timing and swing should be approximately the same, so the place where the ball is and where one wants the ball to be. This can significantly change how one would hit the ball.

4.5 iPact

Following is an explanation of the system using the iPACT method as described in section 3.7. There after, a use scenario is described, detailing how the interaction with the system could be.

Intention

To allow tennissers to acquire feedback and insight on how they tennis as well as giving tennis trainers additional insight and ways to visually explain their feedback to their trainee.

People

Mark (24 years) has been playing tennis since the age of 10. Currently he plays tennis twice a week at the local tennis association. Mark has quite an affinity with technology and usually buys interesting technology quite fast. Therefore Mark also owns the Myo armband.

Peter (38 years) has been a tennis trainer for 15 years. Peter is interested in the new technologies coming up for tennis which allow him to better analyse how someone is playing and allows him to gain a better general overview. The technology should also help him in providing detailed and better feedback towards the tennis player.

Activities

The tennisser will wear the Myo during the tennis practice and will activate the associated application. The camera of the device should be pointed towards the user, such that a video can also be recorded. During the training the user can get a quick insight in their power and movement in the past hits. After the training the user can gain a total overview of how the courses of his movements proceeded, how he hit the ball and other interesting insights.

Context

The application is designed to be used during the training or a match of tennis. The application can be accessed from anywhere to review the past activities which have been stored.

Technology

The system will be using the Myo armband as well as the camera on the phone or computer. Furthermore an application will be used to connect the Myo and the camera.

Use scenario

Mark gets tennis lessons once a week from Peter. During this practise Mark is wearing the Myo, while Peter gives feedback and the application on his telephone is recording the match. At one point Peter gives the feedback to Mark that he should continue his swing more, he is stopping too quickly. Mark does not get what Peter is saying, since he thinks he is properly finishing his swing. At this point Peter grabs the phone and shows Mark how his past swings were done. Now Mark can see in the video footage how he actually stops too soon with his swing.

When looking at the swings Peter notices something else when looking at the visualisation created from the Myo data. Mark is grabbing the racket too tightly, which costs more energy as necessary and deteriorates Mark's play. Peter tells this to Mark and the following swings he looks at the phone more carefully to clearly read-out whether Mark is improving with loosening his arm.

The other tennis session of the week Mark usually plays versus a friend. Now he also wears the Myo and whenever there is a small break Mark quickly goes to his phone to check whether his swings are as Peter tells him. He can do this by both looking at the video footage as well as looking at the Myo data to get a more elaborate overview of what he is doing. At one point in the match Mark aces the serve. Since he usually is not that good in services he wants to know what he did correct this time and therefore quickly runs to his phone and looks at the last hit. He then saves this hit so he can compare it later to other hits, he can then also proudly show this swing later to Peter.

When Mark is done with the match he grabs his phone and stops the match, which automatically is saved. Now the application gives an overview of the match. Allowing mark to see more elaborate statistics on how much he moved and the swings he made. He can also rate this match, so he later knows how well he did this match.

4.6 Preliminary requirements

Out of the above iPACT, use scenarios and interviews some preliminary requirements can be derived. These requirements are based upon the MoSCoW method as described in section 3.8. Furthermore they are divided into functional requirements and non-functional requirements.

FR1: The app must use the Myo

FR2: The app must visualise the movement of the player using the movement data of the Myo

FR3: The app must show the power used by the muscles synchronised with the movement

FR4: The app must store full trainings/matches to be viewed later

FR5: The app should show the movement and power synchronised with a video of the player

FR6: The app should recognise swings and only show the swings, discarding the movement in between

FR7: The app should have the feature of storing separate swings for later viewing

FR8: The app could give an overview of various statistics

FR9: The app could give textual and visual feedback on the swings

FR10: The app won't be built for usage on a smartphone

NFR1: The app must show separate swings within 5 seconds after the swing has been made

As can be seen at FR10 the application won’t be built for usage on a smartphone. This is because the prototype is easier to develop as a computer program, which results in more focus on the other requirements. If the project were to continue the final product would be better as a smartphone application due to its ease of access and portability.

5. Specification phase

The goal of the specification phase is to clearly specify the idea of the tennis trainer in order to get a better overview of the functionalities and system architecture. Finally, this will result in a new set of requirements. This will later be used in the realisation phase to create the application.

5.1 FICS

In section 4.5 a use scenario was given on how the user would experience the interaction with the system. In the coming scenario the same events as in section 4.5 are described, however now the interaction is shown as seen from a system's perspective. For this the FICS method will be used as described in section 3.9.

The first part of the use scenario described Mark training with Peter, where Peter uses the application to further explain his feedback to Mark. This interaction starts with Mark putting on the Myo and calibrating it correctly. This is needed at the start of every exercise. Two parameters are calibrated. First the power Mark uses and second the rotation around the y-axis, since this cannot be determined by the accelerometer. This calibration is done by first starting a new exercise, after which a screen is shown with details about how to calibrate. The calibration that follows involves moving towards the baseline and pointing straight towards the other side and making a fist. Once the system has recognised the fist the Myo will vibrate notifying the user that the calibration has been completed. The orientation is saved as the positive z-axis, so that the user looks at his swings as if he was standing behind him. The power used by making a fist is used as the calibration for the EMG. Notice that during the calibration phase the general Myo poses are unlocked so the system can recognise the fist. Once the calibration is done the system automatically starts the exercise so no new interaction is needed. If the calibration is for some reason unsuccessful (a fist has been recognised before the correct orientation has been assumed) the system can be re-calibrated by selecting this option on the next screen.

During the training the Myo will send orientation and EMG data to the application, which will convert this in a vector (which, if added together, will recreate a swing) and a force (shown as the color of the swing) needed to show the power and orientation correctly on the screen. These forces and vectors are stored in the memory so Peter has the ability to pause what is shown on the application without hindering the recording itself. The visualisation can be paused, fast forwarded and moved backwards by using the buttons displayed at the bottom. If the application has not been paused the latest swing will be shown, which consists of a combination of the current orientation and previous orientations creating a swing. If possible the swing will also show when the ball has been hit. (If FR6 (recognising swings) has not been implemented the application will show the previous second instead of only showing the previous swing.)

The second interaction described in the use scenario was when Mark was playing versus a friend, mainly using the application for later viewing. The start of the match will be the same as described above, starting with the calibration. Once Mark wants to save his swing he walks up to the application. He then uses the arrow buttons to scroll between swings (or to scroll through time). Once he sees the to-be saved swing he presses the star button, telling the system to save the swing. This will make the system save the orientations and forces of the currently shown swing and putting it into "saved swings" folder for later viewing. When Mark wants to look back at his saved swings he selects "saved swings" from the main menu, which will show a list of all the saved swings. When Mark shows the swing to Peter and Peter wants to look at the swing from a different angle, all he has to do is to swipe over the swing, making the viewing orientation change according to the movement of his swipe.

The last interaction consisted of Mark finishing up his match and receiving a match overview. Once Mark has finished his match he walks towards the application and presses the “stop” button. This first gives a notification asking whether Mark really intends to finalise his current match. If the “no” button is selected the match continues as normal, while if the “yes” button is selected Mark is given the opportunity to rate the match. This rating will be a 1-5 star rating allowing Mark to later easily see which match he felt he played well and which he did not. This notification also gives the opportunity to delete the match, if something went wrong or Mark just does not want to save this match he can click this button resulting in another notification asking whether he is sure. Once Mark has given the match a rating he continues to a post-match analysis, showing the amount of movement, calories burned and other interesting statistics of the match. He can also scroll through his hits. If FR9 (automatically giving feedback) is completed the overview can also show the hits which were rated as the best.

5.2 System architecture

The system architecture will show how various components of the system will interact with each other. The FICS will be used as a basis for the interaction. First, the global overview of the system is shown. Second, the application specifically will be further elaborated upon. Finalising will be a couple of Lo-Fi prototypes, which are used to test various algorithms for the final system. The system architecture will be based upon the method described in section 3.10.

5.2.1 Level 0 architecture

The level 0 architecture gives an overview of the complete system. The level 0 architecture can be seen in figure 15.

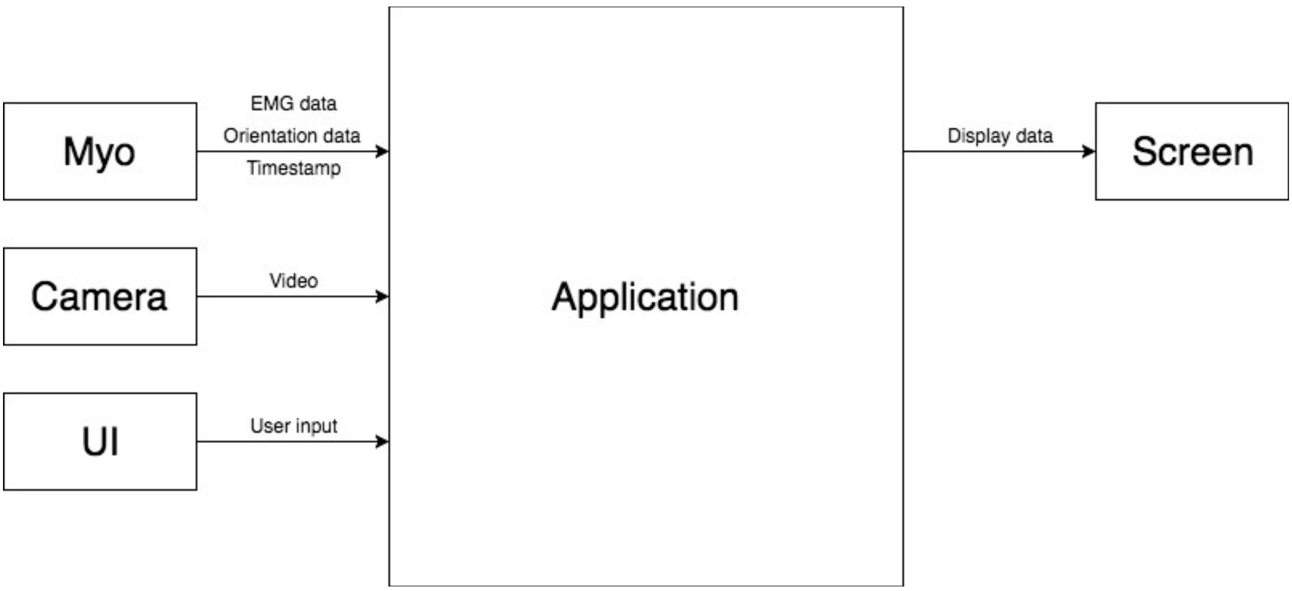


Figure 15: Level 0 architecture

The level 0 architecture consists of three different input devices. First is the Myo, which provides the application with EMG and orientation data as well as a timestamp, which will be used to make sure all data is synchronised. Furthermore, the camera will provide video input, which will be used to complete FR5. Also, the User Interface will allow the user to provide input. This input consists of clicking or dragging across the screen, which will be used to press buttons or change the viewing orientation.

The output of the application will be sent to the display of the screen. This output consists of where to place buttons, the visualisation of the swing and the video. This output will be determined based upon the state in which the program is currently in and which selection is asked by the user to show.

5.2.2 Level 1 architecture

To further explain what is happening inside the application a lower level, level 1, has been created. This level can be seen in figure 16.

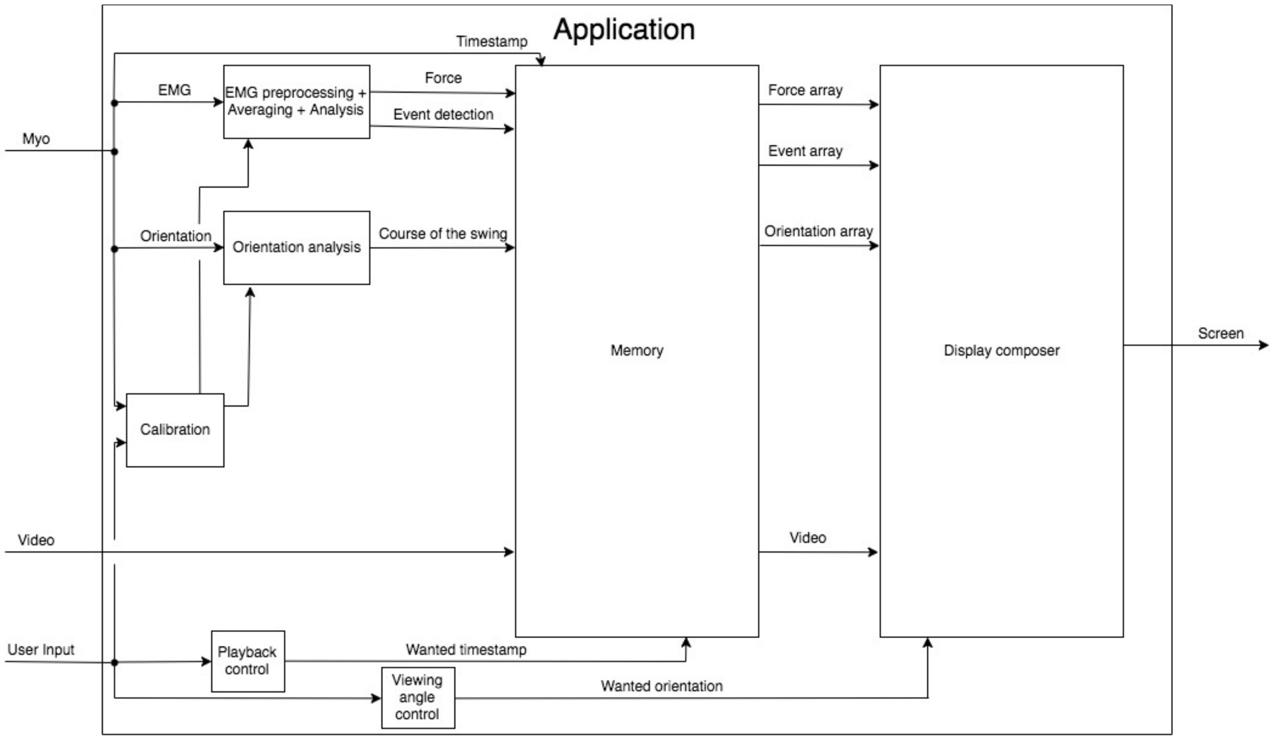


Figure 16: Level 1 architecture

This architecture shows what is happening when the user is recording a match and when he wants to play back. The main menu and such functionalities are left out to improve clarity. This architecture shows how the functionality of the program would look like if FR1 - FR5 are implemented.

Starting on the left side, the same inputs as shown in level 0 can be seen. The Myo signal is divided into four strands. The first being the current timestamp, which is used to synchronise the data by ordering the memory based upon the timestamp. Furthermore the calibration can be seen, which is activated via the user (as told in the FICS) and will then require the fist pose to activate. The orientation and force gained out of this calibration step will then be send to the EMG and orientation preprocessing and analysis.

The EMG preprocessing consists of first rectifying the signal, after which the eight samples from the eight sensors will be added together. The signal is then averaged over a number of previous samples. The amount of previous samples will be determined via the system described in section 5.2.4. There after, the difference with the calibration force is determined to acquire the color with which the power will be represented.

An event detection can also be seen. This is the event of hitting the ball. If the system described in 5.2.4 shows that it can be determined when the ball has been hit, this output will be a boolean which describes whether the ball has been hit or not.

The orientation data is acquired in the form of a quaternion. Quaternions are used to describe a rotation from a certain point. The orientation analysis consists of converting this quaternion gained from the Myo to a vector which can be used as display coordinates. This is done by first rotating the axis of the current rotation to be the same as the display coordinates (the coordinate system of the Myo uses the z-axis up, while the display uses y-axis up), after which the coordinates are shifted according to the calibration. The resulting vector is then stored in the memory.

The video is directly saved into the memory since it does not require any alterations. The video is saved with the latest timestamp given by the Myo as its own timestamp. If tests show that the quality of the video prevents the system from running at the wanted speed, a component to downscale the video can be added to reduce the amount of data that has to be saved.

Finally, the user input can be seen to point towards three components. The first being the calibration, which has already been discussed. The second is towards the playback control, which saves the timestamp that is currently being viewed. The user input will change this timestamp by adding or subtracting, which is done by selected the backwards or forwards buttons. The third user input controls the angle at which the swing is viewed from. This works similar to the wanted timestamp as in that the user changes an already existing value instead of completely inputting its own value. This viewing angle can be changed by dragging over the view of the swing, rotating the image according to the distance and angle dragged.

The output of the memory block is based upon the wanted timestamp. The orientation and EMG data sent is the wanted timestamp as well as the previous measurements such that the final input will be up to one second in the past allowing the display composer to show the movement of the arm in the given second. The video footage will just be the footage present at that timestamp only requiring one instance.

The display composer will then add these variables together as well as adding buttons for the user interaction based upon the current state of the application. The orientation variables can still be changed by the wanted orientation.

5.2.3 Save samples architecture (Lo-Fi prototype)

In order to determine the wanted variables for the various algorithms a set of samples is needed. These samples are acquired using the “save samples” Lo-Fi prototype. This Lo-Fi prototype is shown in figure 17.

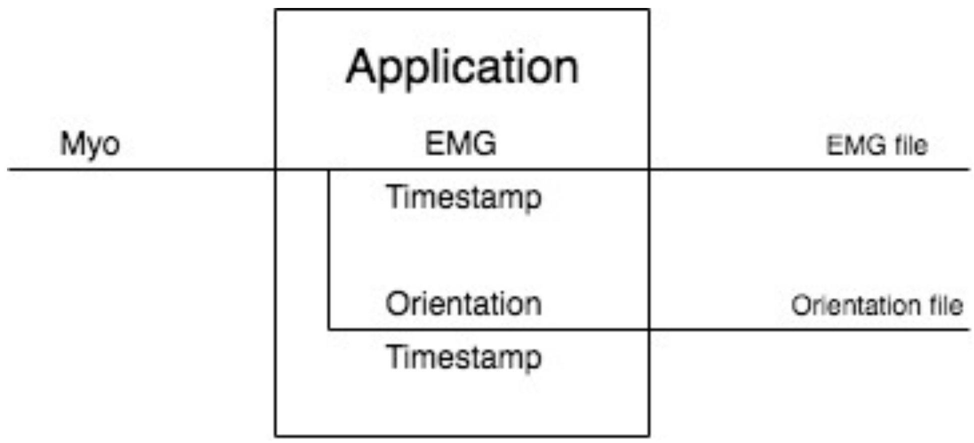


Figure 17: Save samples architecture

This application is rather simple as it only saves the raw EMG and orientation data. The reason to not save the data after the algorithms as in the level 1 architecture is so this data can be used to create those algorithms.

5.2.4 Read out from save architecture (Lo-Fi prototype)

To use the samples acquired by the previous application another Lo-Fi protorype is needed. The goal of this application is to create the needed algorithms. By having the inputted data remain the same the difference between various algorithms and variables can easily be spotted. This application is seen in figure 18.

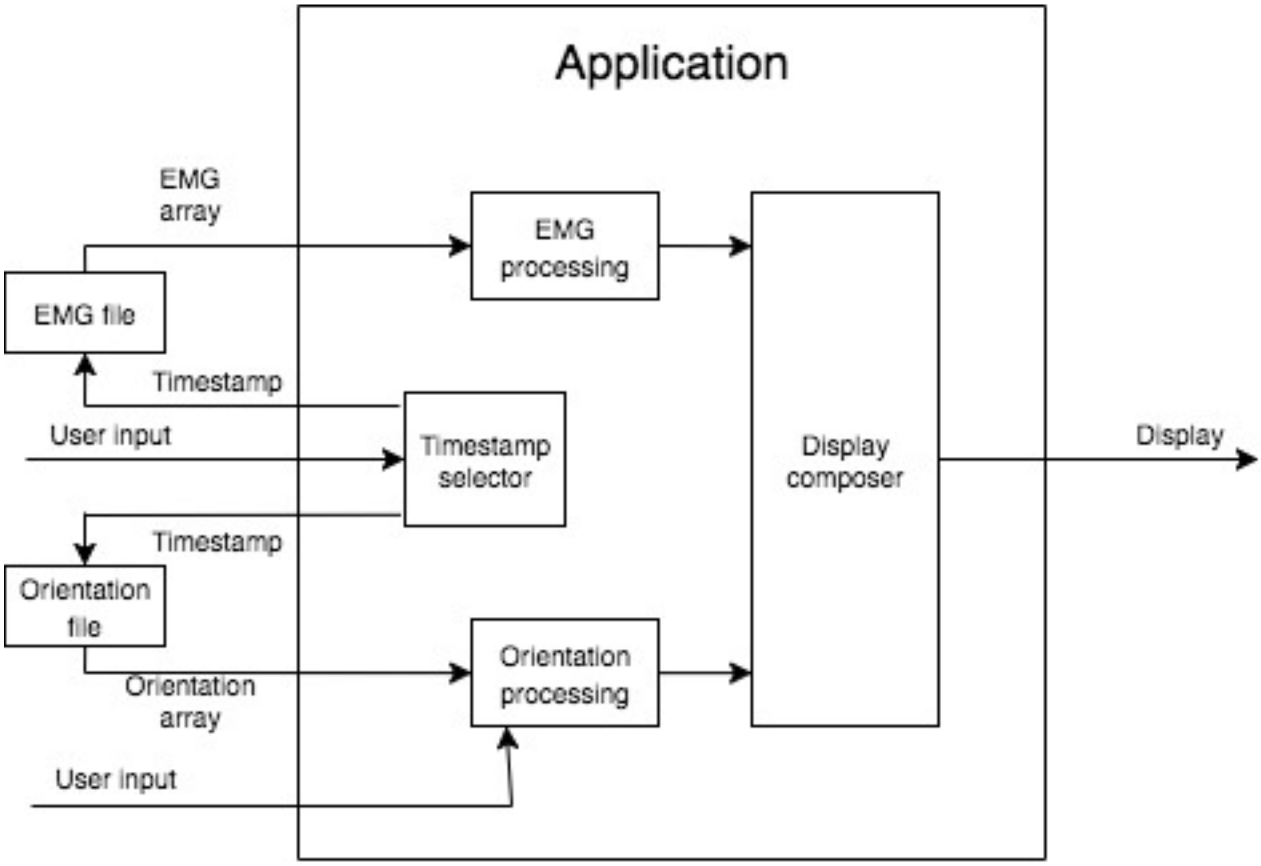


Figure 18: Read out from save architecture

The application starts with the timestamp selector. If nothing is selected this selector will just play through time, showing the samples as if they were inputted at that moment. The outputted timestamp can be changed by the user in two ways. Either the user inputs a wanted timestamp which will then be displayed or the user increases/decreases the speed, pauzes or lets the program run backwards in the EMG and orientation files.

The timestamp outputted by the selector will then be sent to the data samples, which were created using the save samples prototype. This will output an array of data consisting of the current timestamp and enough samples to go one second back into the past, just like what would happen during regular usage.

The processing components are the reason for this program. These will be altered during the testing phase to acquire the best algorithm. The wanted variables and algorithms from the EMG processor are the best amount of samples to use for the Moving Average Filter. Whether the data can show if the ball has been hit and how this can be measured. As well as how to best calibrate the device.

The orientation will mostly be used to test the viewing angle, this is why the orientation processor also has user input. Hitting of the ball might also be visible through the orientation data, which will result in an algorithm testing for this in the orientation processor as well.

The outputs of the composers will go to a display composer. This composer is mostly the same as the one mentioned in figure 16. With the biggest difference being in the difference in user interface and the lack of video footage.

5.3 Requirements

In section 4.6 a set of preliminary requirements has already been defined. This section will therefore elaborate on those requirements also combining those requirements with new requirements which followed out of the FICS and system architecture. The requirements will follow the same methods as before as mentioned in section 3.8.

Must have

FR1: The app must use the Myo

FR2: The app must visualise the movement of the player using the movement data of the Myo

FR3: The app must show the power used by the muscles synchronised with the movement

FR4: The muscle power and movement data must be calibrated in order to match the player and correctly store the orientation.

FR5: The app must store full trainings/matches

FR6: The app must be able to show the stored trainings/matches

FR7: The user must have the ability to pause, play, fast forward and go backwards in the recordings.

NFR1: The app must show separate swings within 5 seconds after the swing has been made

Should have

FR8: The app should show the movement and power synchronised with a video of the player

FR9: The app should recognise swings and only show the swings, discarding the movement in between

FR10: The app should have the feature of storing separate swings for later viewing

FR11: The app should recognise and show when the ball has been hit

FR12: The app should be able to change the viewing angle of the swings

NFR2: The app must have a proper menu allowing the user to easily select what they intend to do

Could have

FR13: The app could give an overview of various statistics

FR14: The app could give textual and visual feedback on the swings

Won't have

FR15: The app won't be build for usage on a smartphone

6. Realisation phase

During the realisation phase the application defined in the specification phase has been created. This has been done by first defining the programming tools to be used. This is followed by the creation of the Lo-Fi prototypes “Save samples” and “Read out from samples”. These Lo-Fi prototypes are evaluated and improvements are made for the Hi-Fi prototype. There after, the Hi-Fi prototype has been created. To finalise the realisation phase the result of the final prototype is shown.

6.1 Programming language

For the creation of the prototypes two different languages were considered: Processing and C++. Below the difference between the two languages is explained. Finally, the choice for the programming language to continue with is made.

6.1.1 Processing

“Processing is a flexible software sketchbook and a language for learning how to code within the context of the visual arts.” [58].

Processing is an easy visuals-based programming language based upon the Java language. It has a third-party Myo library which contains most functions. The main benefit of processing is that it does not require a third-party library for the visualisation of the application and a lot of experience has already been gained in the usage of Processing. The downsides are that the Processing program does not have a good debugging feature and the Myo library does not have access to the onOrientationData function (the function which allows access to the quaternion data), which probably will be used for visualisation the movement.

6.1.2 C++

C++ is a powerful language that allows the creation of a lot of different applications [59]. The main Myo framework by Thalmic Labs is build for C++, which means C++ has access to all the functions related to the Myo, which is also its main benefit. Another benefit for C++ is that it will be built using Xcode, a powerful programming application on the Mac with a lot of debugging tools. The two main downsides of C++ is that the regular output is to a console instead of a visual application for which a third-party framework (OpenFrameworks) is required. This is also linked to the second downside, which is that there is limited experience with the combining of two frameworks. This can lead to the application not compiling if done improperly.

6.1.3 Chosen programming language

For the creation of the application the C++ language is chosen. This is because it has access to all the functions for the Myo and because the downsides only apply to the setup of the application. Once both frameworks are properly linked the downsides of the C++ language will not pose a large problem anymore.

6.2 Lo-Fi prototypes

To better understand how various parts of the program will work a set of Lo-Fi prototypes were created. The created Lo-Fi prototypes are the ones described in chapter 5.2.3 and 5.2.4. After the prototypes were created a functional test was done with these prototypes in order to see in what places the prototypes could be improved when implementing into the Hi-Fi prototype.

6.2.1 Save samples prototype

The code of the “save samples” prototype can be seen in Appendix E. Note that for all code in the appendices only the source files (.cpp) are included since the header files do not provide any extra information. Since this is a Lo-Fi prototype various alternatives for various components were tested, there is a possibility that some parts of these tests can still be seen in the code.

The difference with the architecture in chapter 5.2.3 with the “save samples” prototype is that the setup is not mentioned there, which consists of connecting with the Myo and opening the files in which the application will save the data. The data was saved whenever the hub received new data and all the data was saved.

This prototype would work correctly if the output consisted of two files, one for the EMG and one for the orientation, which both would not have any gaps in between the data.

6.2.2 Read out from save prototype

The code for the “Read out from save” prototype can be seen in Appendix F.

This prototype is based upon the system architecture described in chapter 5.2.4, although there are some differences. First the timestamp selector has less functionalities, the only option added here was the pause option. In order to change this the data has to be loaded differently. Currently the data which is visualised is the only data loaded into the program. Since it is hard and intensive to go back in a read file, it is suggested to load all the data into an array at once allowing the user to go back within that array.

Second change is the way the orientation is changed. This is not done within the orientation processing, but within the display composer, where the user rotates the matrix in which the visualisation is displayed.

This prototype would work correctly if the files as created by the previous prototype were visualised correctly. This means that the orientation is as it was when playing tennis as well as the power. Furthermore, the user should be able to rotate around the visualisation as well as being able to pause it.

6.2.3 Functional testing

In order to test whether the Lo-Fi prototypes were working a couple of functional tests were performed. These functional tests consisted of playing tennis while the “save samples” prototype was running. During these functional tests the location of the bluetooth receiver was changed multiple times in order to see what the best place for the receiver would be in order to get the most data.

During the functional tests one main issue surfaced, which was that given the current setup only half of the packets send by the Myo were received. This resulted in gaps within the recording. It was noted that the application lost connection with the Myo if the user was either at least 7 meters away or was blocking the signal with his body. For this reason the receiver was placed on the side of the field on the side of the dominant arm of the user. The exact location can be seen in figure 19.

Another problem that surfaced due to the packet of data loss was that the visualisation could get very erratic when skipping through the lost sections, resulting in a swing looking faster as it was actually performed. Therefore in the Hi-Fi prototype the data will not be saved when receiving new data but with a sampling rate of 50 Hz. When data is lost the previous data is saved, which results in the visualisation pausing when no new data has been received.

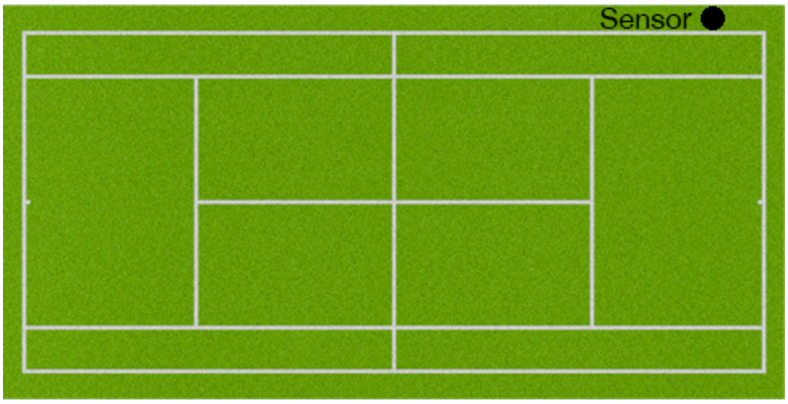


Figure 19: The sensor location which proved best during testing

Using the data from the functional testing the amount of samples for the moving average filter of the EMG was determined. This was based on a visual interpretation of the swings. The choice was made to use 8 samples (0.04 seconds) to average. This resulted in sudden high peaks not having as much of an influence while still being quick enough to register short increases of power.

The functional tests also brought to light another issue. Because tennis involves quick movements the Myo has a possibility of shifting position. If this shift is relatively small this will result in a small notification vibration. If the shift is larger the calibration of the Myo has to be redone. Which results in longer vibrations of the Myo until it is calibrated again. Since the application does not use the poses provided for the Myo it does not require the Myo to be calibrated. Unfortunately, it is impossible to turn off the notifications of the calibration.

The Myo also gives a short vibration when unlocking and locking. This is done by the finger tap gesture. Luckily this functionality can be turned off by using the application in presentation mode, which keeps the Myo in an unlocked state and therefore stops these vibrations.

Since the functionality tests involved users playing tennis, they were also asked about the usage of the Myo during tennis. Most of the users stated not being hindered by the Myo and to not be distracted by it aside from the occasional vibrations. One participant stated that the Myo was on too tightly hindering his movement. The amount of users however was too little to provide any statistical evidence. Since the goal of these tests was to test the functionalities no other questions regarding the application were asked.

6.3 Hi-Fi prototypes

For the Hi-Fi prototype the lessons learned from the Lo-Fi prototypes were implemented and some other changes were made to the system. The code can be seen in Appendix G. The final prototype will be discussed using an updated system architecture. There after, a new functional test has been done to see whether the changes provided any improvements. Finally, the Hi-Fi prototype wil be shown using various visualisations.

6.3.1 Final system architecture

Some changes to the original architecture have been made. These changes were made because of findings from the Lo-Fi prototype, additional/removed features and results found during the programming of the application. Therefore to see how the final prototype works a new architecture has been created which can be seen in figure 20.

There are four differences from the original architecture. The removal of the video input, the separation of memory and current workout data, the addition of the favourite feature and the calibration have been changed.

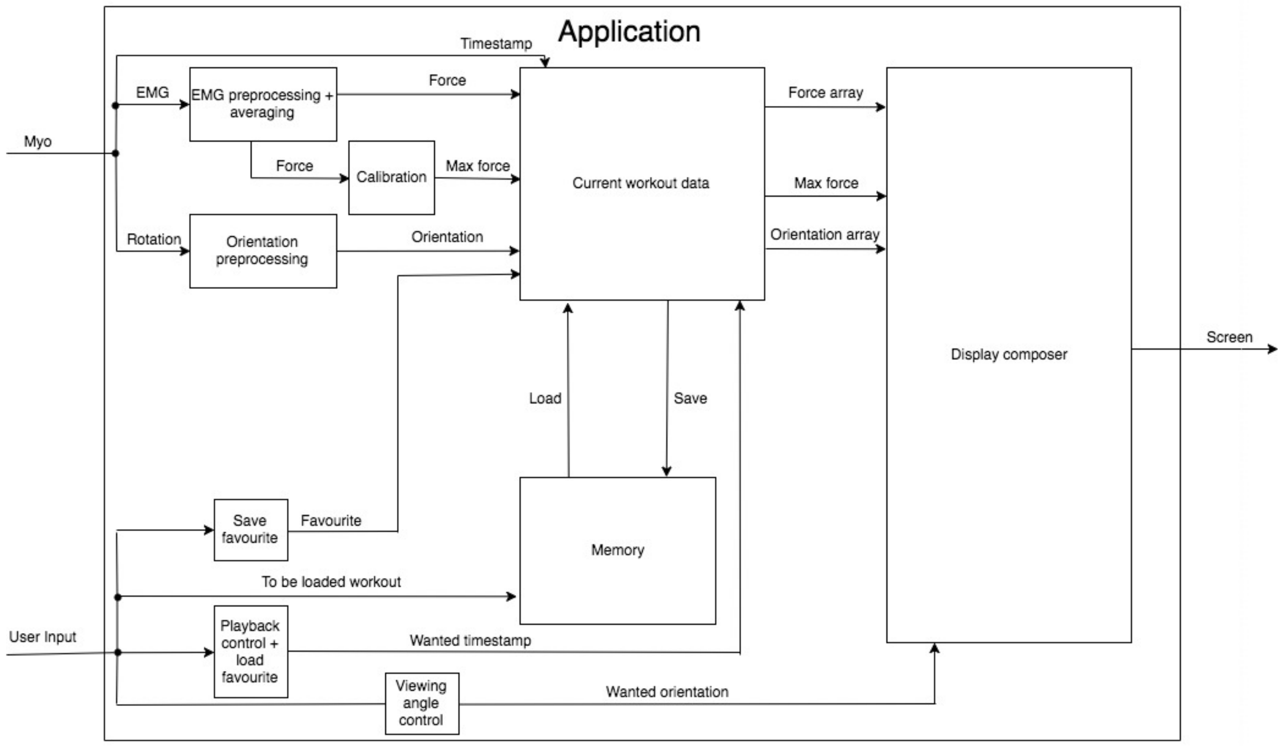


Figure 20: Hi-Fi prototype system architecture

The video input has been removed since it turned out to be too difficult to implement within the time allocated for the creation of the prototype. The main difficulty lied within first acquiring the video footage after which separate frames had to be accessed to be linked to the correct Myo data. Furthermore, it was unknown in what data type the video had to be saved in order for this to be correctly implemented.

The memory was separated from the current workout data to allow easier control of what time is currently displayed. If the display came straight from the memory it would be difficult to go back in time, since you would have to read backwards in the file. Now the data is continuously saved during a new workout. Whenever the workout is finalised the favourites are appended to the end of the file. Whenever a new workout is loaded the data array is immediately filled with the data within the document. Once the loading is done the programs starts the visualisation. This loading has never taken more than a second with the longest workout being 12 minutes long.

The functionality of favouriting a certain time has been added to the application. This feature works as follows: whenever the user is viewing a swing which he wants to save he clicks the “add favourite” button, which adds the current integer value of the data being displayed to an array of favourites. Whenever the user selects a favourite which is to be viewed the data to be displayed is then set to that saved integer value.

The calibration originally required a user input to be performed and also would calibrate the orientation. Although the calibration has a separate page from the rest of the workout it does not require any user input to work. The calibration continuously checks what the highest force emitted is. The currently displayed force is evaluated regarding this maximum force. The displayed color is therefore a percentage of the maximum force. The orientation is not calibrated anymore, since the Myo itself has a drift around the z-axis. This would result in the calibration not being correct anymore after a certain amount of time. Therefore this calibration has not been implemented.

6.3.2 Visualisations

Following is the end result of the application as visualised by screenshots of the application. In figure 21 a tree structure of the program can be seen. Some of the main functions are also mentioned here. The figures are also displayed in a larger format in Appendix H.

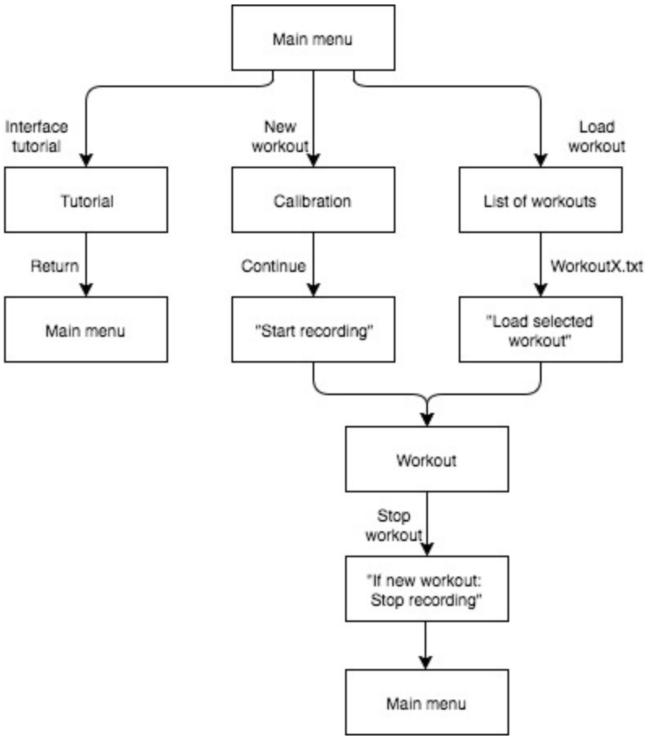


Figure 21: Tree structure of the program

Main menu

The main menu can be seen in figure 22. The main menu links to the three main parts of the application, “Interface tutorial”, “New workout” and “Load workout”. When “New workout” is selected the user first goes to the calibration page before the new workout is started.

Interface tutorial

The interface tutorial can be seen in figure 22. This tutorial is meant to give a brief overview of all the controls during the workout. The interface tutorial does this by an image of a workout where are the buttons are shortly explained with text. All the text is given at once.

Calibration

The calibration page can be seen in figure 23. On this page the user is first prompted to put on the Myo with the status LED towards the wrist. Then the user has to perform the Myo calibration gesture (wave outwards) to prevent the Myo from vibrating throughout the workout. Afterwards the user has to clench his fist tightly to set a high initial maximum force. When done the user should click continue to start the workout.

Workout

The workout page is shown in figure 23. In the middle of the page the swing is seen. The color indicates the force while the direction as seen from the middle is the orientation the Myo was in. If the user wants to see the visualisation from a different angle the mouse should be dragged over the visualisation which rotates it according to the dragging motion. The buttons are used as follows: 1) Add a favourite to the list. 2) Go up the favourite list (a down button can be seen when

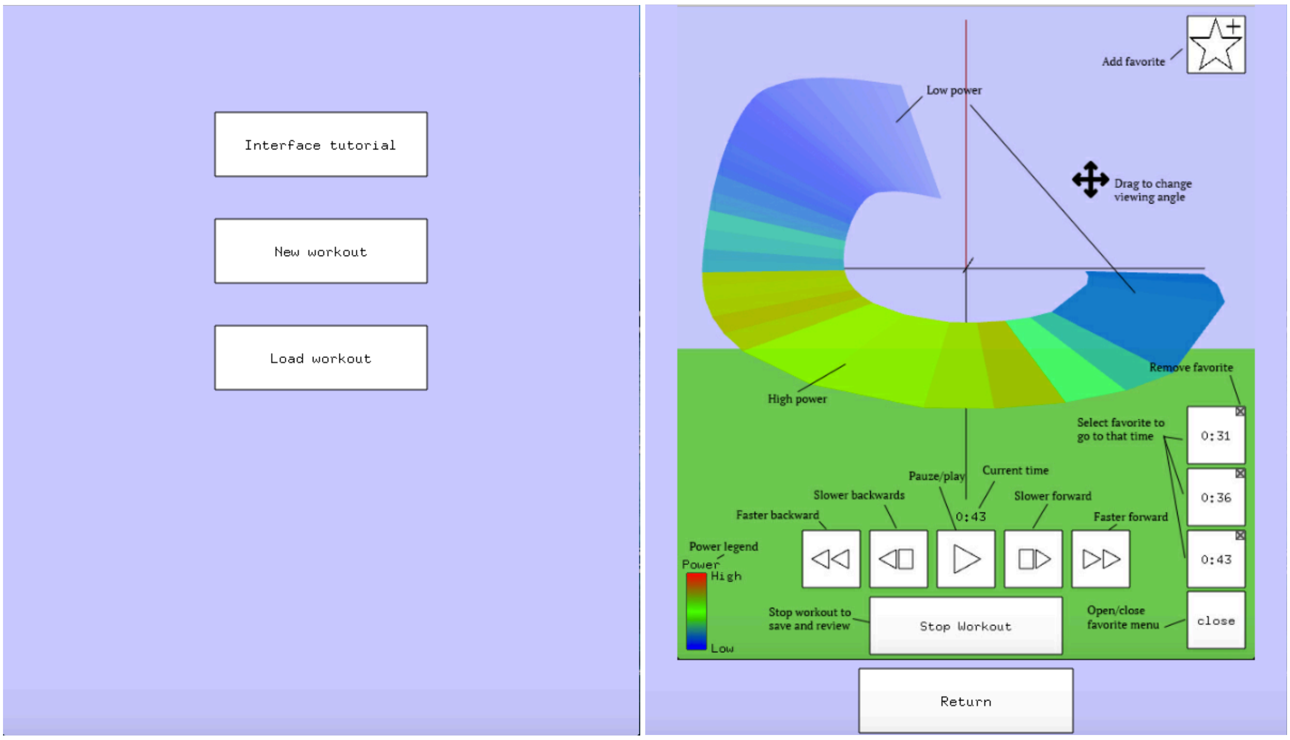


Figure 22: Main menu (left) and interface tutorial (right) of the Hi-Fi prototype

this button is pressed to return) 3) List of favourites, click on one to go to that time. 4) Close the favourite menu. 5) Stop the workout (shows a notification to prevent accidental stops). 6) Legend of the power. 7) Go backwards faster, pressing this button multiple times increases the speed at which the visualisation goes backwards. 8) Go backwards slower, pressing it multiple times decreases the speed at which the visualisation goes backwards. 9) Pause/play. 10) Go forwards slower. 11) Go forwards faster.

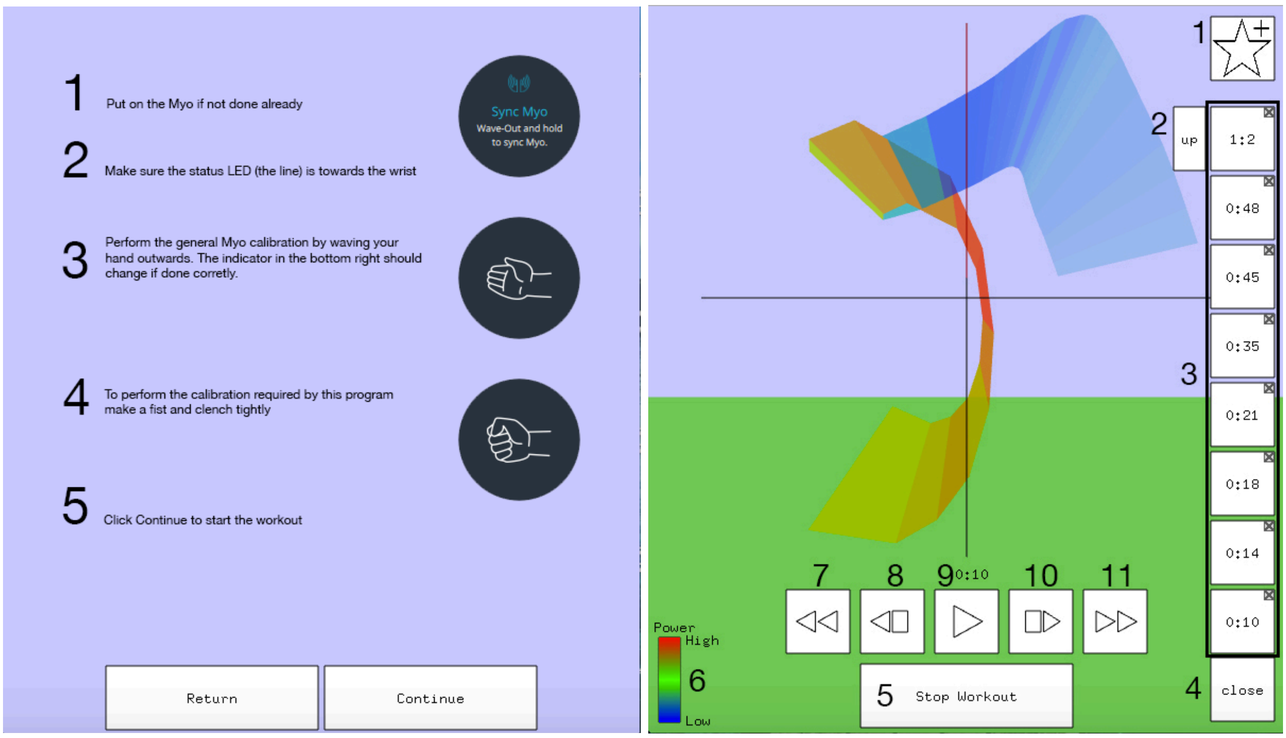


Figure 23: Calibration (left) and workout (right) of the Hi-Fi prototype

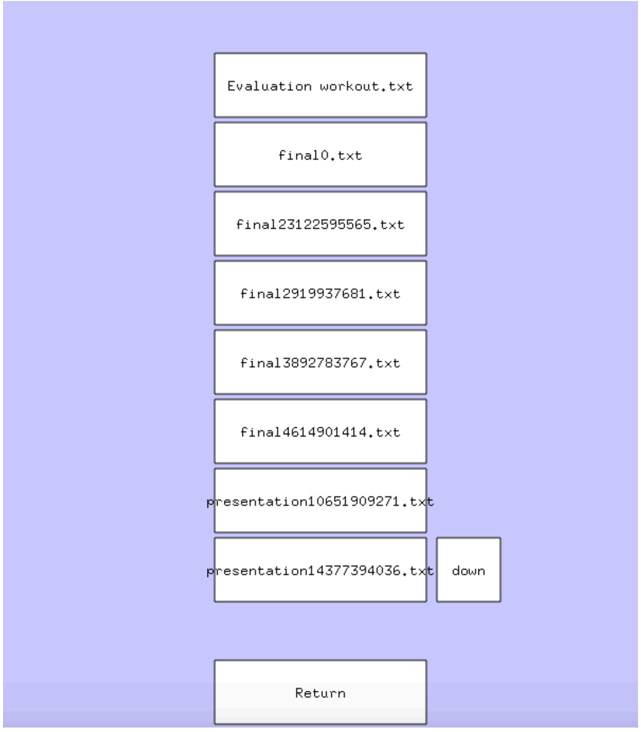


Figure 24: Loading workout of the Hi-Fi prototype

Loading workout

Finally the loading workout page can be seen in figure 24. The workouts shown here can be clicked on which loads the corresponding file into the "current workout data". This will take the user to the workout page reviewing the selected workout. Note that you are now watching an old recording and therefore no new data is being added.

6.3.3 Functional testing

In order to test whether the Hi-Fi prototype was working and whether it provided any improvements compared to the Lo-Fi prototypes another functional test was performed in the same way as the previously done tests.

There were two hypotheses for decreasing the packet loss, which was happening during the first tests. The first hypothesis was regarding the saving of data, which was done at a way higher rate within the Lo-Fi prototype as within the Hi-Fi prototype. This is because the Lo-Fi prototype saved all the data provided by the Myo, resulting in 41 variables being saved every sample, while the Hi-Fi prototype only saves 5 variables for each sample (using the same sampling speed). This can be done since the Hi-Fi prototype first processes the data before saving, which was done the other way around with the Lo-Fi prototype. The second hypothesis was that the bluetooth receiver was too close to the ground resulting in the signal not being received because it was absorbed by the ground. To prevent this the receiver was placed on a chair placing it about 30 centimeters above the ground.

This functional testing was done with both of the above mentioned changes. Which resulted in a large reduction in package loss, which was now at an unnoticeable level. Since both hypotheses were tested at once so it is unknown which one resulted in this decrease in package loss.

Now the package loss was reduced to a minimum it proved to be a lot easier to evaluate the differences between users. Figure 25 shows a forehand performed by two different users. To aid in the recognition of a forehand in figure 26 a forehand is shown. It can be clearly seen that the right forehand is performed better. This can be seen by the force being more steadily built up and built down, resulting in one peak and the swing has a bigger rotation, meaning that this swing was finished properly. Both of these properties are necessary qualities of a proper swing as stated by the interviewed trainer mentioned in section 4.4.

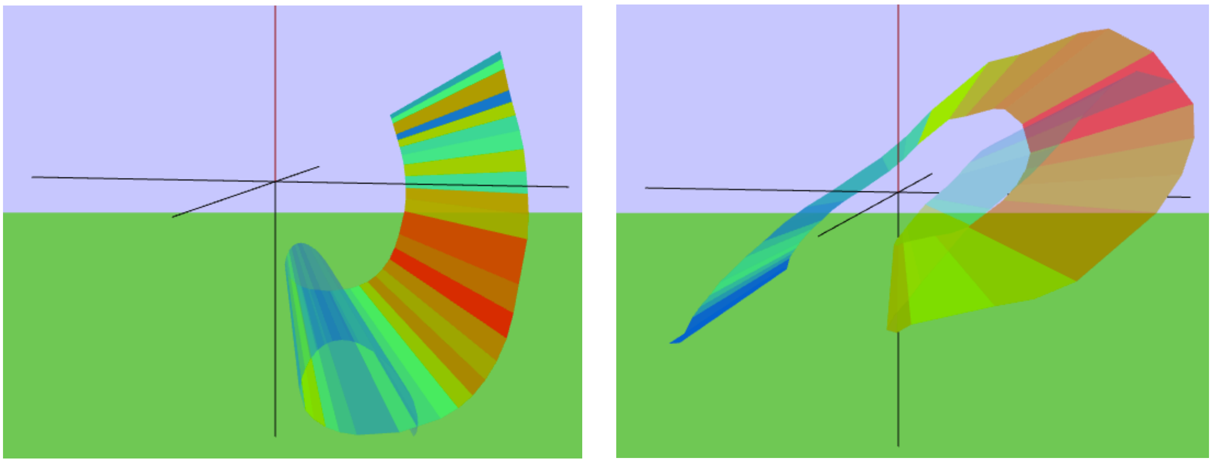


Figure 25: Visualisation of two forehands performed by two different users

Figure 26: Visualisation of a forehand swing

7. Evaluation phase

The goal of the evaluation phase is to test the prototype created in the realisation phase. This evaluation has been done according to the method described in chapter 3.11. First the test procedure is being discussed. Following is the evaluation of the requirements created in the specification phase. Some additional features and requirements came forward from the user evaluation, which are also discussed here. Finally, a discussion on the results presented here will finalise the evaluation phase.

7.1 Test procedure

To properly test the application a script was created in order to keep the information towards the users consistent. This is necessary to allow for consistent measurements across different users. The script involved the setup before the participant would join, acquiring the participant, short description of the evaluation, the evaluation and finalisation.

The setup started by starting a new survey and filling in the next participant number. Further, the application had to be set to main menu. After, the consent form and assignments had to be placed in front of the laptop for the next participant. Furthermore, the final assignment consisted of recognising various swings. To make sure there would not be a learning curve these questions were asked in a random order. This order had to be selected beforehand as well. Finally, the Myo was cleaned.

The participants were acquired from the local study hall and thus all participants were students at Creative Technology within the age range of 18-25. Both users and non-users of the Myo were asked to participate as well as people with and without tennis experience. This was done in order to see whether this would help in the understanding of the visualisation and recognition of the swings. In total a number of 21 people participated in the user evaluation.

The short description involved a short explanation of the Myo and the built application. The only things told about the application were that it involved a tennis trainer and that it visualised the movement and force of the hand. After the explanation the users were asked to fill in the consent form (which can be seen in Appendix I) and to perform the tasks written down in the assignments (which can be seen in Appendix J).

During the evaluation the user had to finish three assignments and was asked to finish the assignment before reading the next one. The first assignment involved creating a new workout. Here the user also had to (pretend to) play tennis. The second assignment was based upon the first where the user had to recognise its own workout. The third workout involved an existing workout. The goal of this assignment was to see whether users were able to recognise different swings not made by themselves. After each assignment a survey had to be filled in regarding the features which were used by the participant during the assignment. The results of this survey will be discussed in the next section. During the third assignment the user was asked to identify five different swings (which were the same across all participants) for one of these swings the users had to also identify the amount of swings.

Once the evaluation was done the participant was thanked for their time and offered a snack.

7.2 Evaluation of requirements

In this section the results of the survey are discussed. The survey will be discussed according to the requirements stated in chapter 5.3. Afterwards any topics not covered by the requirements are discussed. The full results of the survey can be seen in Appendix K.

7.2.1 General results

In general the users were satisfied with the application, which can also be seen in the overall grade, which is on average a 7.2. During the testing some of the user actually stated they were impressed with the prototype and how it can visualise the movement. The participants said that the application would be best if used by trainers and not by tennis players themselves, due to the lack of qualitative feedback.

Many improvements and additions were given for the prototype with most feedback given on the tutorial and calibration. The most promising aspect of the user evaluation was the feedback on the Myo itself, which was stated to be a logical and helpful choice for tennis feedback.

7.2.2 Requirements

FR1: The app must use the Myo

Most participants stated that they found wearing the Myo rather comfortable, however it was not unnoticeable. Most importantly was that hardly anyone stated to be hindered in the slightest bit by the Myo. As for using the Myo as a feedback system for tennis. It was found a logical and practical sensor for tennis.

This requirement was successfully met.

FR2: The app must visualise the movement of the player using the movement data of the Myo

The participants found their movements rather recognisable. And even though participants stated that the visualisation somewhat difficult to understand, they mostly found it clear and useful. Also from the survey the understandability was rated rather positive, so in general it is still positive.

The comments mostly state a clearer way to recognise what direction the swing is moving when the app is paused, some way of recognising in what orientation the player is (am I looking to the front or the side?) and this could be done with a visual of a human. Lastly comments were made about combining the visualisation with video images to get a better indication of what is happening.

Two relevant quotes are: “There should be something recognisable in the interface of the 'movements'. Something alike a human in the center so you can relate the movements to something” and “It is hard to recognise how the movement relates to the arm. However, the speed and power are easily being recognised.”

This requirement was successfully met.

FR3: The app must show the power used by the muscles synchronised with the movement

Most participants found the power to be clearly visualised and could recognise their power being displayed. Most also found the power useful. However only about 30% correctly knew that the power being measured corresponds to the force of the fingers and hand combined.

The main concern mentioned was that a red power does not necessarily have to mean that it is wrong, however the color red was associated with this. The opposite is also true for green. Therefore it might be beneficial to look into another color spectrum. Furthermore further feedback might be included as to what can be done with the insight of providing more or less power.

A relevant quote is: “Make a clear explanation of power and what the consequences of much/less power are.”

This requirement was successfully met.

FR4: The muscle power and movement data must be calibrated in order to match the player and correctly store the orientation.

The calibration, which has to be done before a new workout can actually be started, can definitely be improved with about 30% of the participants stating that the calibration was rather unclear. However, even though the calibration was not clear enough, none of the participants found the calibration annoying.

Since only the muscle power is being calibrated this requirement is only partly met. This is because it turned out that calibrating the orientation would not give the wanted result due to drift within the Myo.

FR5: The app must store full trainings/matches & FR6: The app must be able to show the stored trainings/matches

Since these two requirements are related they will be discussed together. The response to the saving of a workout and selecting one was very positive. Most people stated that selecting an existing workout was very easy and that it was clear.

The comments were mostly about the name of the workout. The participants stated that they either wanted the ability to change the workout name or change the name into something more recognisable like the date and time of the workout.

This requirement was successfully met.

FR7: The user must have the ability to pause, play, fast forward and go backwards in the recordings.

No questions were asked regarding the ability to pause, play, fast forward and go backwards in the recordings. So it cannot be checked how the participants felt about the controls. There were some comments about the controls, which stated that changing the buttons into a time slider would improve the interaction and visibility. It was seen that participants found it hard to understand the controls. Some said that they thought one of the buttons would be a step back while the other was continuous moving backwards. It was also seen that some participants did not click the buttons multiple times to increase or decrease the speed.

Although there is still room for improvements, this requirement was successfully met.

FR8: The app should show the movement and power synchronised with a video of the player

This requirement unfortunately could not be made into a feature of the application. However some participants did state that they would be interested in seeing the visualisation alongside video footage of themselves. This could be implemented into a future version of the application.

This requirement was not met.

FR9: The app should recognise swings and only show the swings, discarding the movement in between

This feature was also not implemented and no comments were made about discarding “idle” data. However there were comments about swing recognition. So if this feature would be added, perhaps it might not be the best to discard the rest, since the participants were not too bothered about it, but to use the feature as a separation method for gesture analysis.

This requirement was not met.

FR10: The app should have the feature of storing separate swings for later viewing

This feature was added and it was called “favourite”, allowing the user to store a specific time and later going back to this time. This functionality was the worst received with a lot of participants stating that the feature was unclear and somewhat difficult to use. The potential of the feature was recognised, since the participants did state it was useful.

Most of the comments were regarding the lack of feedback when adding a favourite, resulting in saving multiple favourites before noticing that one had to open the menu. It was also difficult to understand what was actually favourited. The workout, the swing or a time? Also the add favourite button is the only button at the top of the screen, being easily overlooked. Finally, some participants noted that they wanted to be able to select a beginning and end of a favourite.

Two relevant quotes for the favourite feature were: “There is no instant feedback while saving a favourite, it would be good to have a small pop up that says favourite is saved.” and “Let the user decide the beginning and ending of a 'favourite' instead of one determined moment”.

Although there is still room for improvements, this requirement was successfully met.

FR11: The app should recognise and show when the ball has been hit

This functionality was not included and none of the participants mentioned such a functionality.

This requirement was not met.

FR12: The app should be able to change the viewing angle of the swings

The participants had a difficulty recognising that this feature was present. About 75% did not find this feature on their own. Since the difference this feature could bring was to be discussed, participants were told about this feature so that about half of the participants used the feature, while the others did not.

There were no direct questions regarding the rotation feature. However, using the questions regarding recognising different swings, shows that the feature has a positive impact in the recognition of the swings. The participants who used the rotation feature recognised on average one swing more correctly as the ones who did not, which is 20%.

This requirement was successfully met.

FR13: The app could give an overview of various statistics

This feature was not included in the prototype, therefore there were no questions regarding this requirement. There was a comment about adding some statistics: “Maybe that the Myo will identify automatically what a forehand, backhand or serve is? Maybe an indicator what an average is for your age, gender or level? So you can compare yourselves to other players and make a competition out of it.”

These statistics would not be like the statistics which were thought of earlier (distance moved, calories burned, etc). But would be a nice feature to increase gamification, which is known to attract and keep users.

This requirement was not met.

FR14: The app could give textual and visual feedback on the swings

The idea of having the application give qualitative feedback was not implemented. However, when asked about additional features for the application it was one of the most stated ones.

Some of the participants stated that they wanted to have a regular explanation of how a forehand or backhand should be performed. This did not have to directly be compared to their swings, but for a beginner to have some instructions. Two ways of adding the feature of giving feedback to one's swings were discussed. The first would be feedback on one's swings, by recognising how it should be and discussing it with some improvements. The other way was to directly compare it to a perfect technique letting the user see the difference between the two swings.

Two relevant comments were: "More knowledge about what would be the right way to actually "perform" a forehand, backhand or serve (especially for people who don't play tennis (yet))" and "It may be good (but also very difficult because everyone plays tennis differently) to have a perfect swing programmed so people can repeat that one."

This requirement was not met.

FR15: The app won't be build for usage on a smartphone

No participant discussed the fact that the application was on a computer instead of a smartphone. This could perhaps be due to the fact that the program was clearly a prototype.

This requirement was successfully met.

NFR1: The app must show separate swings within 5 seconds after the swing has been made

No one discussed the speed at which the application shows the swings. The application easily accomplishes this requirement, since it displays the swings instantly.

This requirement was successfully met.

NFR2: The app must have a proper menu allowing the user to easily select what they intend to do

The participants were rather positive about the menu, stating that it was pretty clear and useful. The survey stated that they were a little less convinced about the extensiveness of the menu. There also were some comments about the menu.

The comments were mostly about the menu used during the workout. The comments stated that it was a cluttered mess of buttons and that it was hard to find one's way back to the main menu. A solution for this was also offered: a quick button with an explanation of the menu.

7.2.3 Additional requirements

Besides the requirements some other things were also tested within the user evaluation. This can be either because this was not made into a requirement or because the participants came up with topics not discussed within the requirements.

The tutorial

To aid in understanding what was about to happen when the workout started a tutorial was created. This tutorial gives a quick overview of the user interface, shortly explaining what every button does. This resulted in a lot of information thrown at the users at once, which could also be seen in the comments and the score for the tutorial, stating that it was rather unclear. However, it was recognised as useful. The opinions about the tutorial being lacking or extensive were very varied.

One participant stated that making important parts more distinctive might help or increasing the font size. Also the legend in the bottom left corner, regarding the power was stated to be confusing. Finally a participant stated that the tutorial is difficult to understand because it is very busy.

A functional requirement linked to this topic would be: The application should have a tutorial explaining how the various interactions during the workout can be used.

The users

Some participants also commented upon the users of the application stating that the application would be best for a trainer, looking at the visualisation while also looking at the tennisser. It was also stated that it might be something for tennis clubs to buy so everyone at that club can use it.

A non-functional requirement linked to this topic would be: The application must be built to be most useful for the trainers of tennissers.

However, since one of the earlier stated requirements is to implement qualitative feedback, this requirement has less value. Since, if qualitative feedback is implemented the need for a trainer to interpret the data is lessened.

Sound

There were also some comments regarding the lack of sound. Some participants noted that the sound could be used as feedback when navigating the application, giving feedback when certain buttons are clicked.

A non-functional requirement linked to this topic would be: Sound could be implemented into the application to aid in the navigation of the application and to provide feedback during the game.

Identifying swings

To check whether the participants were able to recognise the swings and therefore whether the application would fulfil its intention a set of questions were asked. These questions asked of the participant to identify various swings. The order in which these questions were asked were different for every user in order to prevent learning from being able to identify certain swings better. The graphs representing the results can also be found in Appendix K.

The graphs show that there is hardly any relationship between the type of swing and the difficulty of identifying one. The participants also did not improve during the questions. However, the only way they could improve was by relating the different swings to each other. Since the answers were not told in between there was no learning curve. The results of these questions can also be seen in figure 27. In this figure the percentage of correct answers can be seen. First, based upon what swing was asked. Second, based upon which question it was. In the second graph it can clearly be seen that no learning curve is present.

The relationship between tennis experience and the average amount of correct identifications was also looked into. It can be seen that people with tennis experience scored higher on average,

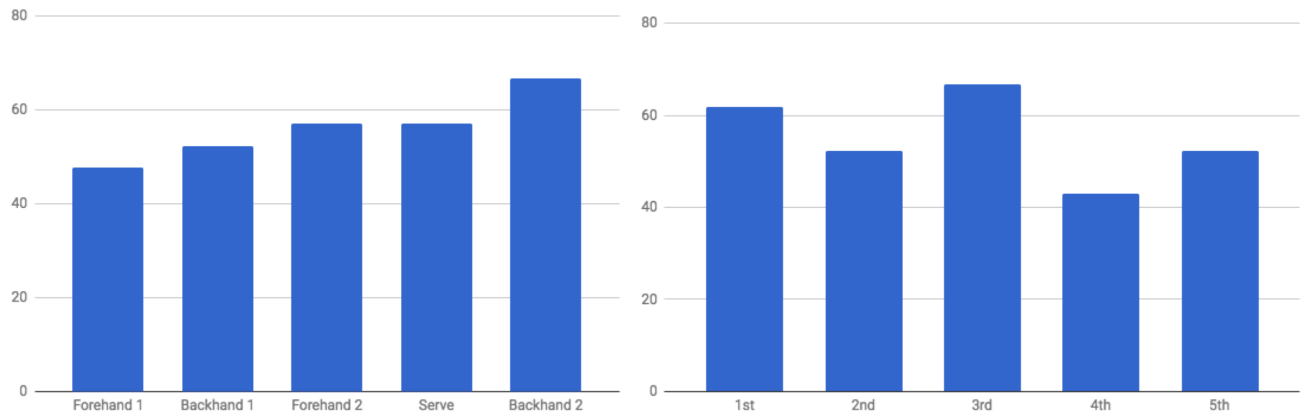


Figure 27: Results of the questions regarding the identification of swings, either based upon the swing type (left) or upon which question it was (right)

however it is yet unknown whether this has any statistical significance. The same can be said about Myo experience and the amount of correct identifications, where the people with Myo experience also scored higher.

When asked about identifying the swings most participants stated that it was rather difficult and somewhat unclear.

The participants were also asked to identify the amount of swings by having to select the fourth swing in the set. 18 out of the 21 participants did this correctly. When asked about recognising the amount of swings the participants were a lot more positive, stating that it was rather easy and clear.

A non-functional requirement linked to this topic would be: The users should be able to identify the amount of swings.

Another non-functional requirement linked to this topic would be: The user should be able to identify the type of swing.

Tennis

The participants were asked about what they thought about the application when used for tennis. The participants were positive about the implementation of this application for tennis, stating that it was rather helpful, could provide an addition and would be effective for the sport.

7.4 Discussion

In general the users were satisfied with the application, which can also be seen in the overall grade, which is on average a 7.2. The most promising aspect of the user evaluation was the feedback on the Myo itself, which was stated to be a logical and helpful choice for tennis feedback.

The user evaluation brought a lot of possible improvements and additions to light. The features which can be improved most are the tutorial and the favourite feature. The tutorial can most be improved in clarity, which can be solved by making a multiple page tutorial allowing the users to take in the information one after another. The main improvement point for the favourite feature lies in adding feedback when adding a favourite and to improve the understanding of what the favourite feature actually does.

The main goal of the user evaluation was to test the requirements stated in chapter 5.3. Now the requirements will be restated with the improvements added by the user evaluation. Note that also the requirements which are not met are restated here, since they might be implemented in a later version.

Must have

FR1: The app must use the Myo

FR2: The app must visualise the movement of the player using the movement data of the Myo

FR3: The app must show the power used by the muscles synchronised with the movement

FR4: The muscle power data must be calibrated in order to match the player.

FR5: The app must store full trainings/matches

FR6: The app must be able to show the stored trainings/matches

FR7: The user must have the ability to pause, play, fast forward and go backwards in the recordings.

NFR1: The app must show separate swings within 5 seconds after the swing has been made

Should have

FR8: The app should show the movement and power synchronised with a video of the player

FR9: The app should recognise swings and only show the swings, discarding the movement in between

FR10: The app should have the feature of storing separate swings for later viewing

FR11: The app should recognise and show when the ball has been hit

FR12: The app should be able to change the viewing angle of the swings

FR13: The application should have a tutorial explaining how the various interactions during the workout can be used.

NFR2: The app must have a proper menu allowing the user to easily select what they intend to do

NFR3: The users should be able to identify the amount of swings within a workout visualisation.

NFR4: The user should be able to identify the type of swing within a workout visualisation.

Could have

FR14: The app could give an overview of various statistics, such as calories burnt or distance moved.

FR15: The app could give textual and visual feedback on the swings

FR16: Sound could be implemented into the application to aid in the navigation of the application and to provide feedback during the game.

Won't have

FR17: The app won't be build for usage on a smartphone

8. Conclusion & Future work

To conclude this Graduation Project the research questions will be discussed once again as well as the answers given within this project. Furthermore the possible future work regarding the created application will be discussed.

8.1 Conclusion

The assignment given at the start of this assignment was to create an innovative application for the Myo. In order to do this, research was done into the Myo as well as comparable devices. There after, an application idea was formed, which was created and tested later on.

In the beginning of the project a series of research questions were asked. During the State of the Art research the first research question "In what domain(s) can the Myo provide actual benefit?" was answered.

To answer this question three subquestions were stated. The first "What are current applications the Myo is being used for?" was answered with a research into the current applications. The five identified fields were Mouse and keyboard replacement, Virtual reality, Entertainment, Robotics and Medical.

To further explore in what domains the Myo could be used a second subquestion was asked "What are possible applications for the Myo which currently do not exist?". This question was answered with a research into related technologies and the applications related to these technologies. This resulted in the domain of Health, which encompasses both Medical as well as Sports, of which the latter was not explored much for the Myo yet.

Finally, the last subquestion "What is the added benefit the Myo provides?" was asked in order to see what benefits can be used in order to create an application in one of the found domains. This question was also answered by looking into related technologies. This time the difference in technological aspects were identified. This resulted in four unique aspects, handsfree usage, large range, battery life and the ability to read out muscle activity.

The second research question "What application can be built so that the Myo actually provides this benefit?" built further upon the first research question. The answer given within this project is a tennis trainer application. This application gives insight into the movements and power of the tennis player. This can be saved for later viewing. Furthermore, the ability to pause, play, go forward and backward into the visualisation as well as rotating around was implemented.

To test whether the application proved successful a user evaluation was done. 21 people participated within these user tests. During the evaluation a set improvements and additions were identified, which will be discussed further in the future work. Some outstanding results were, that the participants found the Myo a logical and helpful choice for tennis feedback. The participants also stated that the tutorial and calibration were the parts which could be improved the most. The final grade given for the application was a 7.2.

8.2 Future work

As already mentioned in the conclusion the user evaluation resulted in a set of improvements and additions for the created application. The most important ones will be reviewed here,. However, for a full insight into possible improvements and additions chapter 7 should be read.

The main improvements for the current application is to first provide better and more feedback. For instance, the favourite feature currently does not provide any feedback whether a favourite has been added. Also the calibration does not provide any feedback whether it has been done correctly.

Furthermore the visualisation should be improved in clarity. This can be done for instance by a human representation, which allows the user to see how the swing relates to his own body. Another way this can be improved is by an arrow indicating which way the swing is moving, which will also be visible when the visualisation is paused.

Finally the current application can be improved by changing the workout names to the format date - time instead of the current naming. This will greatly improve the readability of the workouts. Another option here is to allow the users to alter the workout names.

Besides improvements to the application the user evaluation also brought forward additional features which can be included in future releases. Most of these features have been also mentioned within the should have and could have requirements. First is the addition of video footage linked to the visualisation. This addition would greatly improve the understanding of the visualisation making it easier to improve at tennis.

Second is the recognition of swings by the application itself. If the application can recognise the swings for the user the application becomes easier to use for new users, since it is clearer what they are looking at. This recognition also opens up the way for the final addition.

The final addition mentioned in the user evaluation is qualitative feedback. Once the swings are identified by the application, the application should also be able to compare this swing to how the swing should be. This difference can then be given back to the user in the form of feedback on where to improve.

References

[1] Thalmic Labs, “Myo armband”, 2013. [online] Available: https://www.myo.com/

[2] Thalmic Labs, “Myo marketplace”, 2013. [online] Available: https://market.myo.com/

[3] Thalmic Labs, “Myo armband developerblog”, 2013. [online] Available: https://developer.thalmic.com/

[4] Z. Abraham, D. B. Kwon, T. Solomon, M. Xie, K. Yeh. "Control of an affordable hand and wrist prosthesis." *15th Research Symposium, Rutgers School of Engineering*. 2015.

[5] J. Cheney, and A. Davis. "Gesture controlled virtual reality desktop." 2014.

[6] Huffington Post, “The Myo armband wants to kill your computer mouse”, 2014 [online] Available: http://www.huffingtonpost.com/2014/07/14/armband-motion-controls-computer-mouse_n_5585574.html

[7] The Register, “Meaningful gesture: Thalmic Labs Myo motion sensing armband”, 2015 [online] Available: https://www.theregister.co.uk/2015/10/12/review_thalmic_labs_myo_motion_armband/

[8] Chelmyers, “Myo gesture testing pt. 01”, 2015 [online] Available: http://chelmyers.com/research/2015/06/20/myo-gesture-test-pt-01.html

[9] Thalmic Labs, “Myo armband techspecs”, 2013. [online] Available: https://www.myo.com/techspecs

[10] M.H.B Banierink “Feasibility of ambulant monitoring of arm rehabilitation in stroke patients with the Myo.” Department of Biomedical Systems and Signals, University of Twente, 2016

[11] Adafruit, “Myo armband teardown”, 2016. [online] Available: https://learn.adafruit.com/myo-armband-teardown/inside-myo

[12] Thalmic Labs, “Myo armband diagnostics”, 2013. [online] Available: http://diagnostics.myo.com/

[13] Thalmic Labs, “What are the different LED status descriptions for the Myo armband?”, 2013. [online] Available: https://support.getmyo.com/hc/en-us/articles/202724369-What-are-the-different-LED-status-descriptions-for-the-Myo-armband-

[14] Thalmic Labs, “Myo armband SDK”, 2013. [online] Available: https://developer.thalmic.com/docs/api_reference/platform/the-sdk.html

[15] A.L.P Madushanka, R.G.D.C Senevirathne, L.M.H Wijesekara, S.M.K.D Arunatilake, K.D Sandaruwan. "Framework for Sinhala Sign Language recognition and translation using a wearable armband." Advances in ICT for Emerging Regions (ICTer), 2016 Sixteenth International Conference on. IEEE, 2016.

[16] Thalmic Labs, “Getting to know the SDK”, 2013 [online] Available: https://developer.thalmic.com/docs/api_reference/platform/the-sdk.html

[17] T. Mulling, and M. Sathiyanarayanan. "Characteristics of hand gesture navigation: a case study using a wearable device (MYO)." Proceedings of the 2015 British HCI Conference. ACM, 2015.

[18] L. Zhiyuan, C. Xiang, L. Qiang, Z. Xu, Z. Ping. "A hand gesture recognition framework and wearable gesture-based interaction prototype for mobile devices." IEEE transactions on human-machine systems 44.2, 2014

[19] Z. Xu, C. Xiang, L. Yun, L. Vuokko, W. Kongqiao, Y. Jihai. "A framework for hand gesture recognition based on accelerometer and EMG sensors." IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 41.6, 2011

[20] R. Xu, Z. Shengli, J. L. Wen. "MEMS accelerometer based nonspecific-user hand gesture recognition." IEEE sensors journal 12.5, 2012

[21] G. Pomboza-Junez, and A. Holgado-Terriza Juan. "Control of home devices based on hand gestures." Consumer Electronics-Berlin (ICCE-Berlin), 2015 IEEE 5th International Conference on. IEEE, 2015.

[22] W. Jeen-Shing, and C. Fang-Chen. "An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition." IEEE Transactions on Industrial Electronics 59.7, 2012

[23] S. Mitra and T. Acharya. "Gesture recognition: A survey." IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 37.3, 2007

[24] S. Vorapojpisut, K. Hillairet, A. Boriboonsak, P. Misa. "A Myo Armband-based Measurement Platform for Hand Rehabilitation Applications." Proceedings of the international Convention on Rehabilitation Engineering & Assistive Technology. Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre, 2016.

[25] M. Rutgers "Developing a full feature Myo application." Creative Technology, University of Twente, 2016

[26] LEAP Motion, "LEAP Motion", 2017. [online] Available: https://www.leapmotion.com/

[27] PubNub, "Motion-controlled Servos with Leap Motion & Raspberry Pi", 2015 [online] Available: https://www.pubnub.com/blog/2015-08-19-motion-controlled-servos-with-leap-motion-raspberry-pi/

[28] Microsoft, "Kinect", 2017. [online] Available: http://www.xbox.com/en-US/xbox-one/accessories/kinect

[29] Fitnect Interactive, "Fitnect", 2014. [online] Available: http://www.fitnect.hu/

[30] Microsoft, "Kinect for Xbox One", 2017 [online] Available: http://www.xbox.com/en-US/xbox-one/accessories/kinect

[31] L. Paredes-Madrid, and P. G. De Santos. "Dataglove-based interface for impedance control of manipulators in cooperative human–robot environments." *Measurement Science and Technology* 24.2, 2013

[32] Manus VR, "Manus VR", 2017. [online] Available: https://manus-vr.com/

[33] R.M. Satava. "Virtual reality surgical simulator." *Surgical endoscopy* 7.3, 1993

[34] Android Headlines, "Manus VR Gloves Add A New Dimension To The HTC Vive", 2016 [online] Available: https://www.androidheadlines.com/2016/05/manus-vr-gloves-add-new-dimension-htc-vive.html

[35] Fitbit, "Fitbit", 2017. [online] Available: https://www.fitbit.com/home

[36] Fitbit, "Fitbit Alta HR", 2017 [online] Available: https://www.fitbit.com/altahr

[37] Apple, “Apple Watch series 2”, 2017. [online] Available: https://www.apple.com/lae/apple-watch-series-2/

[38] Google, “Android Wear”, 2017. [online] Available: https://support.google.com/androidwear/answer/6312406?hl=en

[39] The Apple Google, “Top 5 things the Apple Watch 2 must have”, 2016 [online] Available: http://theapplegoogle.com/2016/08/apple-watch-2-3/

[40] A. Mader and W. Eggink, “A design process for creative technology.” In: E&PDE 2014 16th International conference, Enschede, 4-5 September 2014

[41] Mind Toold Editorial Team, "Brainstorming: Generating many radical, creative ideas," in MindTools. [Online]. Available: https://www.mindtools.com/brainstm.html

[42] J. M. Hender, T.L. Rodgers, D.L. Dean, J.F. Nunamaker. "Improving group creativity: Brainstorming versus non-brainstorming techniques in a GSS environment." System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on. IEEE, 2001.

[43] B. F. Crabtree and W. L. Miller, eds. *Doing qualitative research*. sage publications, 1999.

[44] B. DiCicco‐Bloom and B. F. Crabtree. "The qualitative research interview." Medical education 40.4, 2006

[45] R.E. Freeman. “Strategic Management: A stakeholder approach”, Pitman, Boston, 1984.

[46] J. M. Bryson. "What to do when stakeholders matter: stakeholder identification and analysis techniques." Public management review 6.1, 2004

[47] H. Sharp, A. Finkelstein and G. Galal. "Stakeholder identification in the requirements engineering process." Database and Expert Systems Applications, 1999. Proceedings. Tenth International Workshop on. Ieee, 1999.

[48] S. Olander and A. Landin. "Evaluation of stakeholder influence in the implementation of construction projects." International journal of project management 23.4 2005

[49] A. Cooper. “The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity.” Indianapolis, Ind.: Sams, 1999.

[50] N. Larburu, I. Widya, R. G. A. Bults, H. J. Hermens, C. Napolitano. "Early phase telemedicine requirements elicitation in collaboration with medical practitioners." *Requirements Engineering Conference (RE), 2013 21st IEEE International*. IEEE, 2013.

[51] D. T. Ross, and K. E. Schoman. "Structured analysis for requirements definition." IEEE transactions on Software Engineering 1, 1977.

[52] S. Hatton. "Early prioritisation of goals." International Conference on Conceptual Modeling. Springer Berlin Heidelberg, 2007.

[53] M. Glinz. "On non-functional requirements." Requirements Engineering Conference, 2007. RE'07. 15th IEEE International. IEEE, 2007.

[54] S. Robertson, and J. Robertson. Mastering the requirements process: Getting requirements right. Addison-wesley, 2012.

[55] I.A. Widya, R.G.A. Bults, M.H.A. Huis in 't Veld, M.M.R. Vollenbroek-Hutten. "Requirements elicitation in a pain-teletreatment trial," Proceedings of the 17th Anniversary IEEE Joint International Conference on Requirements Engineering, 2009.

[56] D.C. Luckham, J.J. Kenney, L.M. Augustin, J. Vera, D. Bryan, W. Mann. "Specification and analysis of system architecture using Rapide." IEEE Transactions on Software Engineering 21.4, 1995

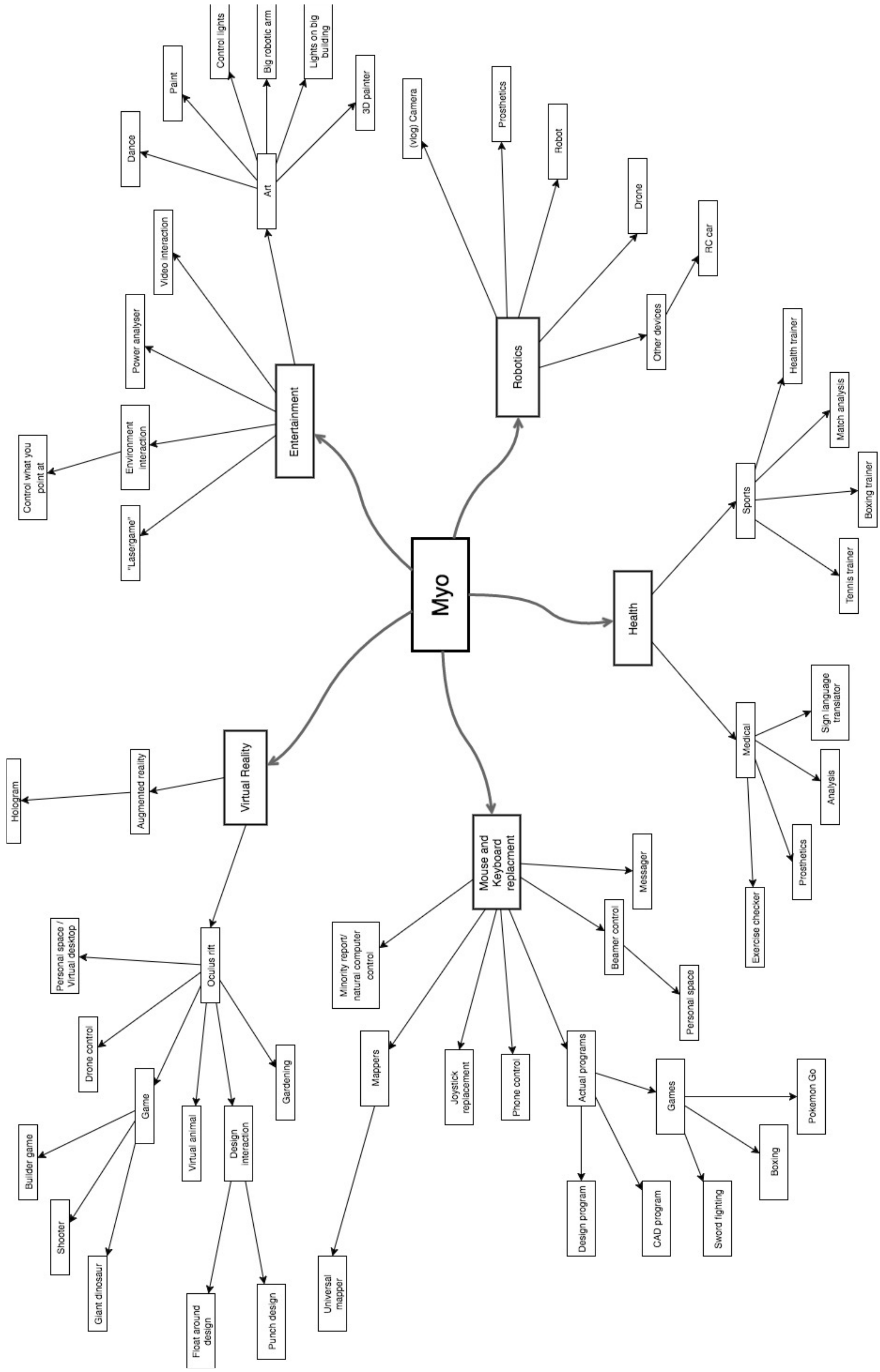
[57] G. Ellis, and A. Dix. "An explorative analysis of user evaluation studies in information visualisation." Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization. ACM, 2006.

[58] Processing, “Processing”, 2017. [online] Available: https://processing.org/

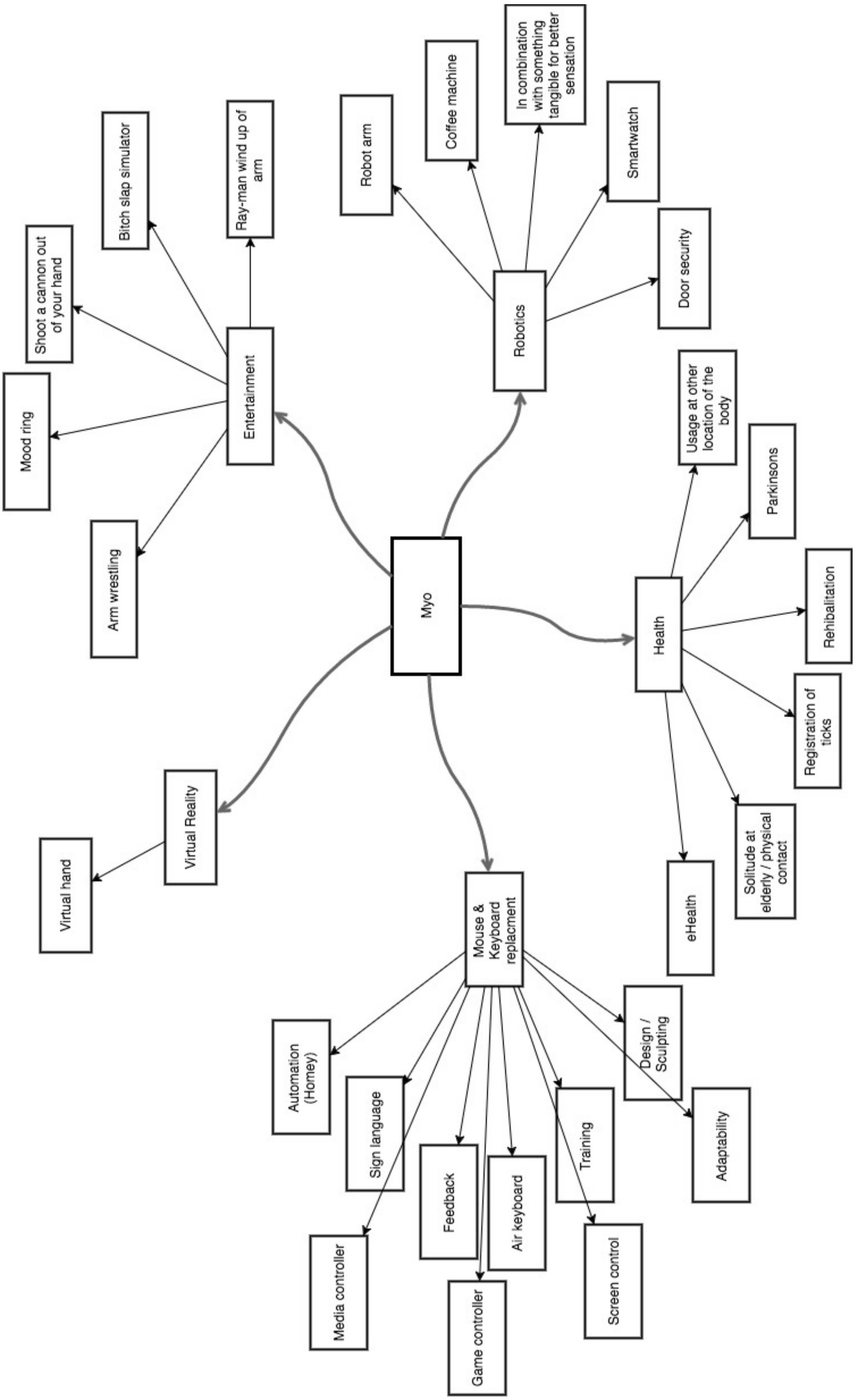
[59] C++, “C++ description”, 2017. [online] Available: http://www.cplusplus.com/info/description/

Appendices

Appendix A: Mindmap



Appendix B: Group brainstorm



Appendix C: Category brainstorm

Add-ons

- On-demand microphone
- Laser pointer
- Rocket system (game of tag)
- Cardboard VR control
- Measuring tape (distance sensor)
- Camera

Serious games

- Rehabilitation
- Conductor
- Designer
- Learn to write
- Health game —> app that is played by exercising

Environment interaction

- Homey
- Draw room (you can draw on all the surfaces around you)
- Light control

Education

- Physical Education
- Powerpoint presentation
- Long distance writing (whiteboard)

Communication

- Messenger
- "Thinking of you"
- Long distance writing (pen control)
- Mood detection

Can I use the Myo for this?

- Navigation
- "Are you awake?" alarm
- Reminders
- Football coaching
- Drink counter
- Virtual guitar

Appendix D: Interview Tennis trainer

Kevin: Ten eerste natuurlijk hartstikke bedankt voor de tijd.

Trainer: Graag gedaan

Kevin: Ik had al in mijn mailtje gezet dat het over de Myo gaat **laat Myo zien** Dit is waar het over gaat. Dit is een apparaat die je zo om je arm doet en vervolgens kan hij hier overal de spieren uitmeten. Door een aantal slimme algoritmes kan hij nu dus een aantal gebaren herkennen, zoals een vuist maken en je vingers strekken. Wat ook kan is puur kijken, hoe sterk pak je je tennisracket vast. En je kan met de Myo de beweging meten. Er bestaan ondertussen al behoorlijk wat applicaties voor, maar ik zag zelf dat er nog helemaal geen sportapplicaties tussen zaten. Ik zat daarnaar te kijken en ik dacht, wellicht kan ik het ook wel koppelen aan het tennissen. Dat je dus de Myo zou gebruiken om te kijken hoe je beter zou kunnen tennissen. En nu ben ik dan op het punt in mijn project aangekomen waar ik ga kijken wat mensen verwachten bij een tennistrainer, wat vinden ze belangrijk? Ik heb op dit moment dus nog geen applicatie, maar ik ga dit gesprek gebruiken om de basis neer te zetten van hoe het eruit zou moeten komen te zien. Is dat duidelijk?

Trainer: Ja zeker, wat wij wel doen is, wij werken bijvoorbeeld met video. Dat kan een iPhone zijn of gewoon een laptop, daar zit dus een programmaatje op waarmee mensen dan kunnen meekijken naar het zwaaiverloop. Dat soort dingen doen we wel. Maar goed dit is voor ons ook nieuw. Vandaar dat we dus ook geïnteresseerd zijn.

Kevin: Dus jullie staan ook wel positief tegenover het gebruik van techniek in de tennissport?

Trainer: Zeker.

Kevin: De vragen die ik ga stellen zullen voornamelijk open vragen zijn waar een hele hoop antwoorden op mogelijk zijn en waar vooral jou mening naar voren kan komen. Als ik hier een applicatie voor zou maken, wat denk je dan dat belangrijk zou zijn dat je in ieder geval zou willen weten?

Trainer: Nou ja, in hoeverre iemand met heel veel kracht speelt of niet. Hoe verhoudt zich dat. In principe als je tennist heb je natuurlijk dat je indraait en op het moment dat je je racket in de valactie hebt, dan ga je meer knijpen. En de vraag is waar de persoon dan gaat knijpen. En hoeveel knijpt ie? En wanneer ontspant die weer? Dat zijn best wel belangrijke dingen om te weten.

Kevin: En hoe zou je dat dan voor je zien dat je dat dan als feedback terug krijgt?

Trainer: Dat weet ik eigenlijk niet precies hoe dat opgenomen wordt. Als je merkt dat iemand al knijpt op het moment dat hij naar achter gaat kan je aangeven dat het de bedoeling is dat hij daar meer ontspant. Anders kost het wel heel veel kracht en als je alleen je spierkracht gebruikt dan remt het ook meer dan dat het een acceleratie is.

Kevin:Je laat hem vallen juist om de snelheid te maken?

Trainer: Juist, vanuit de val de spierkracht erbij en dan kan je dus harder slaan.

Kevin: Ik zit zelf te kijken of ik het voor de tennistrainers wil maken of voor de tennissers zelf. Want voor de trainer zou het dus heel handig kunnen zijn om te zien wanneer iemand in de spieren knijpt en met je mobieltje ernaast en dat zodra er geslagen is het mobieltje erbij pakken om te zien hoe er geslagen is. Maar stel je bent gewoon een tennisser, je hebt een wedstrijd waar de trainer dan niet bij is, stel je zou dan een applicatie ervoor hebben, dat je na de wedstrijd terug kan kijken. Hoe zou dit er dan uit moeten zien?

Trainer: Dan zou je misschien in combinatie met beeld moeten doen. Als je een beweging van iemand op zou nemen dan zou je dus zodra de beweging start kunnen laten zien dat iemand al begint te knijpen. Daar knijp je bijvoorbeeld te veel of te weinig. Ik denk dat het met het zien wel heel belangrijk is.

Kevin: Dat je dus ergens een camera neerzet en dat je dat dus synchroniseert met hoeveel je knijpt? Op die manier had ik er nog niet naar gekeken.

Trainer: Ik kan me bijvoorbeeld ook wel voorstellen met voetbal dat als je aan het begin van de trap je spieren al helemaal vastzet je ook geen lekkere trap maakt. Als je dan vanuit de zwiep pas je spieren meer aan gaat spannen gaat het een stuk beter. Dat is bij tennis eigenlijk ook zo. Vanuit de valactie ga je versnellen en daarna draait de romp mee en gebruik je ook spierkracht.

Kevin: Nu zit je dan vooral op de spierkracht te kijken, maar hij kan ook de beweging meten. Zou daar wellicht nog applicaties of toegevoegde waarde in liggen?

Trainer: Hoe bedoel je met beweging? Met de zwaai in algemeenheid?

Kevin: Ja, de zwaai in algemeenheid of wat ik eerder had gehoord is dat het belangrijk is dat je helemaal door moet slaan. Dat hij wellicht dus kan meten dat iemand te vroeg stopt.

Trainer: Ja zou kunnen. Ik koppel dat voor mezelf altijd wel aan het zien. Als je bijvoorbeeld van een speler een filmpje zou hebben, van de zijkant of de achterkant, dat je kan laten zien: Kijk daar gebruik je teveel kracht, omdat je daar al stopt. Ze gebruiken vooral veel kracht om eerder te stoppen. Vooral vanuit het ontspannen, gestrekte arm aanspannen naar het weer ontspannen. Heel veel mensen remmen dan op het raakpunt af, omdat ze dan bang zijn dat ze de bal uit slaan of dat ze het gevoel hebben te weinig controle te hebben. Ik zie dat altijd wel. Wat ik ook al zei is dat we ook al veel met de laptop doen op een standaard en dan maken we opnames. Het is voor mensen altijd wel belangrijk om het te koppelen aan het zien. Ik volg veel filmpjes van tennisorganisaties in het binnen en buitenland. Dan kijk je naar wat zei op Youtube zetten. Het slaan, de beweging, een combinatie van een aantal dingetjes. Ik denk dat het belangrijk is dat je het een beetje koppelt. Had je nog meer trainers gesproken dan?

Kevin: Op dit moment nog niet, ik heb wel met daadwerkelijke tennissers gepraat, maar die vinden het belangrijker om later terug te kunnen kijken dan op het moment zelf. Het zijn ook allemaal tennissers die echt in de technische wereld zitten en dus veel meer bezig zijn met wat het apparaat kan en dan kan ik dit dus eruit pakken. Veel minder met wat er wellicht nog bijgezet kan worden, zoals bijvoorbeeld een filmcamera en dan is het veel duidelijker.

Trainer: Ik zou het zelf natuurlijk ook niet zeker weten, ik heb daar ook gewoon mijn beeld bij. Ik ben zelf heel visueel ingesteld. Dat merk ik ook bij veel kinderen die ik op les heb. Kinderen willen voornamelijk zien wat er gebeurd, zien wat er wordt bedoeld, zien wat de beweging is. Als je kijkt naar filmpjes op Youtube, zoek maar op forehand of backhand en dan heb je een hele hoop trainers die iets hebben opgenomen en hun idee erbij hebben gezegd. Dat is heel visueel en sport is natuurlijk ook heel visueel. Er wordt ook altijd gezegd: Oh wat een mooie tennisser of kijk eens naar die forehand van Nadal. En dan komt er weer een heel verhaal bij. Ik ben ook weleens benieuwd naar de spierkracht verhouding. Je zou Federer wellicht zo'n band om kunnen doen en Nadal. Dan krijg je wel heel verschillende data. Nadal doet echt gewoon alles op kracht en heel veel kracht. Dat ben ik wel eens benieuwd als je dat in een grafiek zou zetten en dan het verschil bekijken tussen Federer en Nadal. Met hoeveel kracht beweegt Nadal zijn racket naar de bal en hoe verhoudt zich dat tot andere spelers? Dat is ook wellicht grappig om te weten, maar goed dat dwaalt misschien iets af.

Kevin: Stel, ik zat zelf al over een ideaal beeld te denken. Dat ik dan professionele tennissers de applicatie ook kon laten gebruiken om dan een database op te bouwen van zo hoort het. Dat je wellicht kan terugkoppelen hoe iemand het heeft gedaan en hoe het hoort. Dat je dat dan ook met elkaar kan vergelijken.

Trainer: Dan zou je dus eigenlijk al een referentie van iemand moeten hebben.

Kevin: Vandaar dus de professionele spelers. Die dus goed weten hoe het hoort.

Trainer: Maar er zit wel altijd een stuk beeld bij. Want als jij dus bijvoorbeeld zwaait *staat op en laat het zien*. Doe je eerst je racket naar achteren en zodra je hem laat vallen, zwaai je hem naar voren. Als ik de bal bijvoorbeeld hier raak *met de arm recht naar voren gestrekt* kost het mij al veel meer kracht dan als ik hem hier raak *met de arm wat losser, zodat het meer overeen komt met de valbeweging*. Als ik dan niet een beeld erbij heb dan zegt die spierkracht nog niet zo heel veel. Dan kan je dus zeggen dat iemand veel te veel kracht gebruikt, maar het verschil zit vooral in waar de bal wordt geraakt. Als iemand te veel kracht gebruikt maar wel een goed timingsmoment heeft is het al een stuk beter als dat iemand al de verkeerde timing of swing heeft. Dan moet je dus eerst de zwaai aanpassen, dan moet je dus eerst een beeld hebben van een zwaai en van een raakpunt. Dat moet dus overeen komen met wat je hebt opgenomen. Anders heb je dus een heel lastig uitgangspunt.

Kevin: Een van de manieren hoe ik het voor me zag is *laat schets zien* dat de bewegingssensoren gebruikt worden om een baan te visualiseren. Dat je dus geen daadwerkelijk beeld hebt, maar dat je een 3D reconstructie hebt van hoe de baan is verlopen. Dat je dus hier *wijst aan op schets* kan zien dat hij groen is en zodra hij naar beneden gaat dat de spieren aangespannen worden en dan richting rood gaat. Hier heb je dus niet het daadwerkelijke beeld van de speler, denk je dat dat dan nog uit zou maken?

Trainer: Dan moet je wel een uitgangspunt hebben. Dan moet je dus wel van de ideale techniek uitgaan. Als je uitgaat van Federer, zo iemand zou dan je uitgangspunt zijn. Ik denk dat Nadal niet het goede uitgangspunt zou zijn, omdat hij dus al heel veel met kracht corrigeert. Een heel ander timingsgevoel dan Federer. Messi heeft bijvoorbeeld ook een heel fijn gevoel voor timing en een hele fijne motoriek. Dan zou je dus moeten neerzetten: Dit is van Federer en dan leg je die van mij ernaast en dan is dat het verschil. Maar dan moet het uitgangspunt in zwaai en timing wel ongeveer hetzelfde moeten zijn.

Kevin: Wat ik dus dacht is dat dit *wijst naar schets* puur was wat was opgenomen, ook de beweging. Dus dat ik dus, als ik nog tijd over heb, dit als referentie gebruikt zou worden. Maar dat het ook als losstaand iets gebruikt kan worden, dat je dus niet een hele camera setup hoeft neer te zetten, maar dat je gewoon puur de Myo om moet doen, dat je gaat tennissen en dat je daarna gaat kijken. Hoe ik het verwacht te kunnen krijgen is dat ik kan registreren wat een slag is, want als je gewoon beweegt is dat natuurlijk heel anders dan als je een slag maakt. En dat hij dus registreert wanneer een slag begint en wanneer hij eindigt en dat hij dus per slag opslaat. Dat je vervolgens alle slagen op een rijtje te zien krijgt en dat je er zo doorheen zou kunnen kijken.

Trainer: Maar zie je dan ook nog beeld van de persoon? Of alleen de reconstructie?

Kevin: Alleen de reconstructie van de sensoren en dus geen beeld.

Trainer: Maar hoe ziet dat er dan uit? Hoe zou ik dat voor me moeten zien? Een speler denkt wellicht “Ik moet minder knijpen in mijn beweging naar achteren”, dan is dus wel het handigste om direct het beeld erbij te zien. Als je mij bijvoorbeeld aan de bovenkant zou zien en dan de kleur van de kracht mee laat lopen. Dan kan je dus aangeven, op dit punt is hij bij jou al rood, terwijl hij nog groen moet zijn. En als hij dan in de valactie komt dan moet hij dus van oranje naar rood gaan. Dan heb je het beeld gekoppeld aan de sensoren. Anders krijg je alleen dat je kan zeggen “Daar knijp je teveel” en dan krijg je terug “Ja, waar is dat dan?”. Want er is geen beeld bij, toch?

Kevin: Nee, geen daadwerkelijk beeld. Maar het zou inderdaad zo kunnen maken dat je je mobiel tegen een bankje aan zet en dat je vervolgens de camera van je mobiel gebruikt. Dat is ook wel een interactie die mogelijk zou kunnen zijn.

Trainer: want ik denk wel, als je kijkt naar alle filmpjes die er van tennis zijn gemaakt. Hoeveel filmpjes van tennis drills erop staan. Tennis drills zijn oefeningen waarbij ik dan de bal zou spelen op de forehand, backhand, forehand etc dat noemen ze een tennis drill. Hoeveel filmpjes er zijn gemaakt over baseline spel, forehand, backhand, serve, volleys, combinatie. Iedereen, overal ter wereld zet filmpjes erop. De ervaringen, de kennis die zij hebben willen delen met de wereld en of het nou waar is of niet, het is voor hun de waarheid, dus je kan er altijd wat van leren. Je wilt het kunnen zien, of het stilzetten van een beweging. Als ik bij een service beweging hier en hier al knijp *doet een service beweging voor*, dan krijg ik een drukkertje. Als ik hem hier eerst losser laat, waardoor hij in mijn nek valt en daarna mijn spieren aanspan, dan krijg je een zwieper. En door die zwieper, als een lasso krijg je veel meer kracht erachter. Maar dan moet je wel beeld erbij hebben. Als de applicatie dan met het beeld mee loopt, dan kan je tegen een speler zeggen, dat hij op bepaalde punten moet ontspannen. Als hij dan eerder knijpt dan komt er geen snelheid uit en dan is het timingsgevoel ook heel ruw. Moet je maar kijken naar Roland Garros nu, die valactie bij de service. Dan zwiepen ze het racket eruit.

Kevin: Deze is me nooit opgevallen, ik dacht zelf altijd dat het een soort van halve cirkel was.

Trainer: Nee, ze laten hem echt wel vallen in de nek. Ik heb nu geen filmpjes voor de hand. Kijk er maar eens heel specifiek naar. Ik zeg ook altijd tegen mensen die dan meer van het voetenwerk willen leren. Kijk maar eens heel specifiek een wedstrijd alleen naar het voetenwerk. Dus nu dan door die valactie en daarna je spieren aanspannen, daardoor ga je versnellen. Doe je dat te vroeg, dan kom je niet goed uit. Doe je dat te laat dan werkt het ook niet. Dan zit je racket zo los dan gebeurd er niks. Dan zou je dus met de applicatie kunnen zien, hier zet je al teveel kracht en je zit nog maar in de achterzwaai. En je moet hem wel door laten gaan, dus als je ergens ook nog verkeerd knijpt werkt het ook niet.

Kevin: Even kijken of ik voor de rest nog vragen heb. Volgens mij is dit wel redelijk wat ik had.

Trainer: Ik wil best wel een keer afspreken. Mocht je een keer willen dat we een aantal keer wat bewegingen doen. Dan moet je dat maar gewoon aangeven. Kan altijd.

Kevin: Goed te horen, hartstikke leuk.

Trainer: Dat we uiteindelijk naar iets komen waarvan we zeggen ja dit moet het zijn. Maar ik denk wel, maar dat ben ik misschien, dat je het graag zou willen zien. Als iemand tegen mij zegt, wat vind je van deze speler, hij doet dit, dit en dit. Dan zeg ik altijd, graag een plaatje erbij.

Kevin: Hij kan wel iets doen, maar dat kan nog steeds op tien, honderd verschillende manieren.

Trainer: Ik wil er graag bij staan om het te ervaren.

Kevin: Ik wil nu eerst graag de techniek werkend krijgen. Ik kan nu al wel heel simpel meten hoeveel kracht iemand gebruikt. In principe zou ik dus al heel snel iets neer kunnen zetten met een beeld en een kleur erop. Maar ik wil ook wel kijken of dat ook nog met beweging erbij kan, dat je dus ook heel duidelijk terug kan zien, dat je de baan ziet lopen en daar een kleur op.

Trainer: Is ook prima. Ik geef vooral aan vanuit mijn trainerschap, aangezien wij ook veel met beeld werken. Omdat kinderen ook vaak niet helemaal door hebben wat ze dan doen. Als je bijvoorbeeld met de service bezig bent, dan heb je de linkerarm en de rechterarm en dan ben je wellicht met de linkerarm bezig en heb je helemaal niet door wat de rechterarm doet. Dan kan je dus zeggen, laten we even kijken, dan zetten we de fragmenten even naast elkaar en dan heel rustig. Dan zeggen de kinderen vaak “Oh, ik had echt het idee dat ik dat deed”. Dan hebben ze het beeld erbij en dan gaat het vaak wel wat makkelijker.

Kevin: Is het dan heel lastig te zien of iemand heel erg aanspant?

Trainer: Nee, je ziet meestal wel in het verloop. Mensen hebben dan een kortere zwaai of stoppen eerder. Dan zie je wellicht ook dat de knokkels wit zijn. Dan kan je dus zeggen, je moet de onderarm wat meer ontspanning geven, want het is meestal wel de onderarm. Dat is vak wel wat lastig. Dat zie je meestal vooral bij senioren. Junioren die beginnen met een zachte bal zijn vaak wat speels en dat gaat een stuk makkelijker.

Kevin: Ja en die hebben het dan ook vroeg geleerd. Dus dat blijft dan ook in het systeem. Zijn er voor de rest nog vragen? Dingen die onduidelijk zijn? Of nog opmerkingen? Dingen die je kwijt wil?

Trainer: Nee, ik ben vooral heel benieuwd, hoe het uiteindelijk gaat werken. Want er zijn natuurlijk een hele hoop dingen die links en rechtsom worden bedacht. Ze zijn bijvoorbeeld wel heel fel met camera's. Mijn collega in Doorn heeft bijvoorbeeld ook allemaal speedcamera's boven, onder, links en rechts. Dat werkt natuurlijk wel heel verhelderend. Daar gaan wij natuurlijk ook steeds meer mee werken. In combinatie met dan deze methode ben ik vooral heel benieuwd hoe dat eruit komt te zien. En wat het een speler kan brengen, want dat is natuurlijk ook heel belangrijk.

Appendix E: Save samples prototype

ofApp.cpp

```
#include "ofApp.h"
#include "myo/myo.hpp"
#include <iostream>
#include <fstream>

myo::Hub hub("com.create.savefiles");
DataCollector collector;

//--------------------------------------------------------------
void ofApp::setup(){
    // We catch any exceptions that might occur below -- see the catch statement for more
details.
    try {


        std::cout << "Attempting to find a Myo..." << std::endl;

        // Next, we attempt to find a Myo to use. If a Myo is already paired in Myo
Connect, this will return that Myo immediately.
        // waitForMyo() takes a timeout value in milliseconds. In this case we will try
to find a Myo for 10 seconds, and if that fails, the function will return a null pointer.
        myo::Myo* myo = hub.waitForMyo(10000);

        // If waitForMyo() returned a null pointer, we failed to find a Myo, so exit with
an error message.
        if (!myo) {
            throw std::runtime_error("Unable to find a Myo!");
        }

        // We've found a Myo.
        std::cout << "Connected to a Myo armband!" << std::endl << std::endl;

        // Enable the EMG streaming
        myo->setStreamEmg(myo::Myo::streamEmgEnabled);

        // Hub::addListener() takes the address of any object whose class inherits from
DeviceListener, and will cause
        // Hub::run() to send events to all registered device listeners.
        hub.addListener(&collector);

        collector.setup();

    } catch (const std::exception& e) {
        std::cerr << "Error: " << e.what() << std::endl;
        std::cerr << "Press enter to continue.";
        std::cin.ignore();
        //return 1;
    }
}

//--------------------------------------------------------------
void ofApp::update(){
    //The Hub run provides the output for the swing, because it runs at 30 FPS, the whole
application does
    hub.run(1000/30);
}
```

Collector.cpp

```
// Copyright (C) 2013-2014 Thalmic Labs Inc.
// Distributed under the Myo SDK license agreement. See LICENSE.txt for details.
#define _USE_MATH_DEFINES
#include <cmath>
#include <iostream>
#include <iomanip>
#include <stdexcept>
#include <string>
```

```
#include <algorithm>

#include <fstream>
#include <chrono>

// The only file that needs to be included to use the Myo C++ SDK is myo.hpp.
#include <myo/myo.hpp>
#include <ofmain.h>

// Classes that inherit from myo::DeviceListener can be used to receive events from Myo
devices. DeviceListener
// provides several virtual functions for handling different kinds of events. If you do
not override an event, the
// default behavior is to do nothing.
class DataCollector : public myo::DeviceListener {
public:
    DataCollector()
    : onArm(false), isUnlocked(false)
    {
    }

    std::ofstream emgfile, orifile;

    void setup(){
        cout << "How is the emgfile called?";
        string emgfilename;
        cin >> emgfilename;
        emgfile.open(emgfilename);

        cout << "How is the orientationfile called?";
        string orifilename;
        cin >> orifilename;
        orifile.open(orifilename);



        if(!emgfile){
            std::cout << "EMG does not exist";

        } else if (!emgfile.good()){
            std::cout << "EMG cant write";
        }

        if(!orifile){
            std::cout << "Ori does not exist";

        } else if (!orifile.good()){
            std::cout << "Ori cant write";
        }

    }

    void onConnect(myo::Myo *myo, uint64_t timestamp, myo::FirmwareVersion
firmwareVersion) {
        //Reneable streaming
        myo->setStreamEmg(myo::Myo::streamEmgEnabled);
        //myo->vibrate(myo::Myo::vibrationMedium);

    }

    // onUnpair() is called whenever the Myo is disconnected from Myo Connect by the
user.
    void onUnpair(myo::Myo* myo, uint64_t timestamp)
    {
        // We've lost a Myo.
        // Let's clean up some leftover state.
        onArm = false;
        isUnlocked = false;
    }

    // onOrientationData() is called whenever the Myo device provides its current
orientation, which is represented
```

```
    // as a unit quaternion.
    void onOrientationData(myo::Myo* myo, uint64_t timestamp, const
myo::Quaternion<float>& quat)
    {
        outputOri(timestamp, quat);
    }

    void outputOri(uint64_t timestamp, const myo::Quaternion<float>& quat){
        if (!orifile){
            std::cout << " no file";
        }else if(!orifile.good()){
            std::cout << "error with file!";
        } else if(orifile.is_open()){
            std::string output = "";

            output.append(std::to_string(timestamp));

            output.append(",");
            output.append(std::to_string(quat.x()));
            output.append(",");
            output.append(std::to_string(quat.y()));
            output.append(",");
            output.append(std::to_string(quat.z()));
            output.append(",");
            output.append(std::to_string(quat.w()));

            output.append("\n");

            orifile << output;
            std::cout << output;
        } else{
            std::cout << "failed at everything";
        }
    }


    //OnEmgData is called whenever the Myo provides new data
    void onEmgData(myo::Myo* myo, uint64_t timestamp, const int8_t* emg){
        outputEmg(timestamp, emg);
    }

    void outputEmg(uint64_t timestamp, const int8_t* emg){
        if (!emgfile){
            std::cout << " no file";
        }else if(!emgfile.good()){
            std::cout << "error with file!";
        } else if(emgfile.is_open()){
            std::string output = "";

            output.append(std::to_string(timestamp));

            for (int i = 0; i < 8; i++){
                output.append(",");
                output.append(std::to_string(emg[i]));
            }

            output.append("\n");

            emgfile << output;
            std::cout << output;
        }else{
            std::cout << "failed at everything";
        }
    }

    // onArmSync() is called whenever Myo has recognized a Sync Gesture after someone has
put it on their
    // arm. This lets Myo know which arm it's on and which way it's facing.
    void onArmSync(myo::Myo* myo, uint64_t timestamp, myo::Arm arm, myo::XDirection
xDirection, float rotation,
                   myo::WarmupState warmupState)
    {
```

```
        onArm = true;
        whichArm = arm;
    }

    // onArmUnsync() is called whenever Myo has detected that it was moved from a stable
position on a person's arm after
    // it recognized the arm. Typically this happens when someone takes Myo off of their
arm, but it can also happen
    // when Myo is moved around on the arm.
    void onArmUnsync(myo::Myo* myo, uint64_t timestamp)
    {
        onArm = false;
    }

    // onUnlock() is called whenever Myo has become unlocked, and will start delivering
pose events.
    void onUnlock(myo::Myo* myo, uint64_t timestamp)
    {
        isUnlocked = true;
    }

    // onLock() is called whenever Myo has become locked. No pose events will be sent
until the Myo is unlocked again.
    void onLock(myo::Myo* myo, uint64_t timestamp)
    {
        isUnlocked = false;
    }

    // These values are set by onArmSync() and onArmUnsync() above.
    bool onArm;
    myo::Arm whichArm;

    // This is set by onUnlocked() and onLocked() above.
    bool isUnlocked;
};
```

Appendix F: Read out from save prototype

ofApp.cpp

```
#include "ofApp.h"
#include "myo/myo.hpp"
#include <iostream>
#include <fstream>

//DataCollector collector;
std::ifstream orifile, emgfile;

//--------------------------------------------------------------
void ofApp::setup(){
    cout << "How is the emgfile called?";
    string emgfilename = "emgwouter1.txt";
    //cin >> emgfilename;
    emgfile.open(emgfilename);

    cout << "How is the orientationfile called?";
    string orifilename = "oriwouter1.txt";
    //cin >> orifilename;
    orifile.open(orifilename);

    ofSetFrameRate(10);
}


myo::Vector3<float> rotation;
deque<ofVec3f> rotationArray;

ofColor color;
deque<ofColor> colorArray;

// onOrientationData() is called whenever the Myo device provides its current
orientation, which is represented
// as a unit quaternion.
void onOrientationData(uint64_t timestamp, const myo::Quaternion<float>& quat)
{

    myo::Quaternion<float> rotateQuat(-0.5f,0.5f,0.5f,0.5f);
    rotateQuat = rotateQuat*quat;
    rotateQuat = rotateQuat.normalized();
    //std::cout << std::fixed;
    //std::cout <<std::setprecision(2);
    //std::cout << "w:" << quat.w() << "  x:" << quat.x() << "  y:" << quat.y() << "  z:"
<< quat.z() << '\r';
    //std::cout << timestamp;

    rotation = myo::Vector3<float>(300,0,0);
    rotation = rotate(rotateQuat, rotation);

    //std::cout << "         x:" << setw(8) << setfill(' ') << rotation.x() << "  y:" <<
setw(8) << setfill(' ') << rotation.y() << "  z:" << setw(8) << setfill(' ') <<
rotation.z() << '\r';

    rotationArray.push_front(ofVec3f (rotation[0], rotation[1] ,rotation[2]));
    colorArray.push_front(color);
    while(rotationArray.size() > 30) {
        rotationArray.pop_back();
        colorArray.pop_back();
    }
}

// These values store the EMG values
int mav;
int emgData[8];
float emgMax = 200;

void onEmgData(uint64_t timestamp, const int8_t* emg){
```

```
    // Store 10 total values of the emg sensor
    for(size_t j = 0; j < sizeof(emgData)/sizeof(emgData[0]-2); j++){
        emgData[j]=emgData[j+1];
    }

    //insert the new values in new EmgData
    emgData[sizeof(emgData)/sizeof(emgData[0])-1]=0;
    for (size_t i = 0; i < 8; i++) {
        emgData[sizeof(emgData)/sizeof(emgData[0])-1]+=abs(static_cast<int>(emg[i]));
    }

    //Moving average filter
    mav=0;
    for(size_t j = 0; j < sizeof(emgData)/sizeof(emgData[0]-1); j++){
        mav+=emgData[j];
    }
    mav/=(sizeof(emgData)/sizeof(emgData[0]));

    if (emgMax<emgData[sizeof(emgData)/sizeof(emgData[0])-1]) emgMax =
emgData[sizeof(emgData)/sizeof(emgData[0])-1];
    //std::cout << emgMax << '\n';

    if (mav<emgMax/2){
        color.b=255-(mav/emgMax*255);
        color.g=mav/emgMax*510;
        color.r=0;
    } else {
        color.b=0;
        color.g=510-(mav/emgMax*510);
        color.r=mav/emgMax*255;
    }
}

uint64_t readEmg(){
    if(!orifile.eof()){
        string istring;
        uint64_t timestamp;
        int8_t emg[8] = {0,0,0,0,0,0,0,0};

        getline(emgfile, istring, ',');
        std::istringstream iss(istring);
        iss >> timestamp;

        for (size_t i = 0; i < 7; i++) {
            getline(emgfile, istring, ',');
            emg[i] = stoi(istring);
        }
        getline(emgfile, istring);
        emg[7] = stoi(istring);

        onEmgData(timestamp, emg);

        return timestamp;
    } else {
        std::cout << "end of EMG file";
        return 9999999999;
    }
}

bool pauze;
uint64_t timeOriOld = 0, timeOriDif = 0;
long totaltime = -20475872015;
int amount;

void readOri(){
    if(!orifile.eof() && !pauze){
        string istring;
        uint64_t timeOri = 0, timeEmg = 0;

        getline(orifile, istring, ',');
        std::istringstream iss(istring);
        iss >> timeOri;
```

```
        if(orifile.eof()) return;

        float quatx, quaty, quatz, quatw;
        getline(orifile, istring, ',');
        quatx = std::stof(istring);
        getline(orifile, istring, ',');
        quaty = std::stof(istring);
        getline(orifile, istring, ',');
        quatz = std::stof(istring);
        getline(orifile, istring);
        quatw = std::stof(istring);

        myo::Quaternion<float> quat(quatx, quaty, quatz, quatw);

        while(timeEmg<timeOri){
            timeEmg = readEmg();
        }
        timeOriDif=timeOri-timeOriOld;
        //color=timeOriDif/2500;
        timeOriOld=timeOri;
        amount++;
        totaltime+=timeOriDif;

        cout << timeOriDif/1000 << "    " << totaltime/amount/1000 << "    " <<
float(timeOri)/1000000 << "    " << float(timeEmg)/1000000 << '\n';

        onOrientationData(timeOri, quat);
        //cout << timeOri << "    " << quat.x() << "   " << quat.y() << "   " << quat.z() <<
"   " << quat.w() << "   "  << '\n';
    } else {
        //std::cout << "end of orientation file";
    }
}

//--------------------------------------------------------------
void ofApp::update(){
    readOri();
}

int rotateX = 0;
int rotateY = 0;
//--------------------------------------------------------------
void ofApp::draw(){
    //Draw the course of the swing
    ofPushMatrix();
    ofTranslate(512, 384);
    ofRotateX(rotateY/2);
    ofRotateY(rotateX/2);
    if (rotationArray.size()>=1){
        for(int i = 1; i < rotationArray.size(); i++) {
            //Set the color based upon the power of the muscles
            ofSetColor(colorArray[i]);

            //The swing consists of various rectangles, connecting the current value with
the previous one, which creates the visualisation of a swing
            ofBeginShape();
            ofVertex(rotationArray[i].x/2, rotationArray[i].y/2, rotationArray[i].z/2);
            ofVertex(rotationArray[i].x, rotationArray[i].y, rotationArray[i].z);
            ofVertex(rotationArray[i-1].x, rotationArray[i-1].y, rotationArray[i-1].z);
            ofVertex(rotationArray[i-1].x/2, rotationArray[i-1].y/2,
rotationArray[i-1].z/2);
            ofEndShape();
        }
    }

    //These are the axis shown
    ofSetColor(0, 0, 0);
    ofDrawLine(-300,0,0,    300,0,0);
    ofDrawLine(0,-300,0,    0,300,0);
    ofDrawLine(0,0,-300,    0,0,300);
    ofPopMatrix();
```

```
}
//--------------------------------------------------------------
void ofApp::keyPressed(int key){
    if (key == 112 && !pauze) {
        pauze = true;
    } else if (key == 112 && pauze) pauze = false;
}

int xOld;
int yOld;
bool first = true;
//--------------------------------------------------------------
void ofApp::mouseDragged(int x, int y, int button){
    if(first){
        xOld = x;
        yOld = y;
        first = false;
    }
    rotateX+=x;
    rotateX-=xOld;
    rotateY-=y;
    rotateY+=yOld;
    xOld=x;
    yOld=y;
}

//--------------------------------------------------------------
void ofApp::mouseReleased(int x, int y, int button){
    first = true;
}
```

Appendix G: Hi-Fi prototype

ofApp.cpp

```
#include "ofApp.h"
#include "myo/myo.hpp"
#include <iostream>
#include <fstream>

myo::Hub hub("com.create.tennistrainer");
DataCollector collector;
int state = 0;
bool newWorkout;
ofMesh gradient;

//--------------------------------------------------------------
void ofApp::setup(){
    ofSetFrameRate(50);
    // We catch any exceptions that might occur below -- see the catch statement for more
details.
    try {


        std::cout << "Attempting to find a Myo..." << std::endl;

        // Next, we attempt to find a Myo to use. If a Myo is already paired in Myo
Connect, this will return that Myo immediately.
        // waitForMyo() takes a timeout value in milliseconds. In this case we will try
to find a Myo for 10 seconds, and if that fails, the function will return a null pointer.
        myo::Myo* myo = hub.waitForMyo(10000);

        // If waitForMyo() returned a null pointer, we failed to find a Myo, so exit with
an error message.
        if (!myo) {
            throw std::runtime_error("Unable to find a Myo!");
        }

        // We've found a Myo.
        std::cout << "Connected to a Myo armband!" << std::endl << std::endl;

        // Enable the EMG streaming
        myo->setStreamEmg(myo::Myo::streamEmgEnabled);

        // Hub::addListener() takes the address of any object whose class inherits from
DeviceListener, and will cause
        // Hub::run() to send events to all registered device listeners.
        hub.addListener(&collector);


    } catch (const std::exception& e) {
        std::cerr << "Error: " << e.what() << std::endl;
        std::cerr << "Press enter to continue.";
        std::cin.ignore();
        //return 1;
    }

    //load images
    star.load("images/star.jpg");
    cross.load("images/cross.png");
    addstar.load("images/addstar.jpg");
    tutorial.load("images/tutorial.jpg");
    calibration.load("images/calibration.png");

    gradient.setMode(OF_PRIMITIVE_TRIANGLE_STRIP);
    gradient.addVertex( ofPoint(10,ofGetHeight()-10) );
    gradient.addColor(ofColor(0,0,255));
    gradient.addVertex( ofPoint(30,ofGetHeight()-10) );
    gradient.addColor(ofColor(0,0,255));
    gradient.addVertex( ofPoint(10,ofGetHeight()-50) );
    gradient.addColor(ofColor(0,255,0));
    gradient.addVertex( ofPoint(30,ofGetHeight()-50) );
```

```
    gradient.addColor(ofColor(0,255,0));
    gradient.addVertex( ofPoint(10,ofGetHeight()-90) );
    gradient.addColor(ofColor(255,0,0));
    gradient.addVertex( ofPoint(30,ofGetHeight()-90) );
    gradient.addColor(ofColor(255,0,0));

    ofSetBackgroundColor(200, 200, 250);

}

//--------------------------------------------------------------
void ofApp::update(){
    switch (state){
        case 0: // Main menu
            break;
        case 1: // Tutorial
            break;
        case 2: // Calibration
            hub.run(1000/50);
            calibrate(&collector);
            break;
        case 3: // Workout overview
            break;
        case 4: // Show workout
            if (newWorkout) {
                hub.run(1000/50);
                insertData(&collector);
            }
            break;
    }
    //The Hub run provides the output for the swing, because it runs at 30 FPS, the whole
application does

}

int rotateX = 0, rotateY = 0, filesToShow=0, favoritesToShow;
int largeButtonWidth = 200, largeButtonHeight = 60, smallButtonWidth = 60,
smallButtonHeight = 60;
bool showFavorites = false, sureToExit = false;

//--------------------------------------------------------------
void ofApp::draw(){
    switch (state){
        case 0: // Main menu
            //TO-DO insert background movement data as distraction
            button(ofGetWidth()/2-largeButtonWidth/2, 100, largeButtonWidth,
largeButtonHeight, "Interface tutorial");
            button(ofGetWidth()/2-largeButtonWidth/2, 200, largeButtonWidth,
largeButtonHeight, "New workout");
            button(ofGetWidth()/2-largeButtonWidth/2, 300, largeButtonWidth,
largeButtonHeight, "Load workout");
            break;
        case 1: // Tutorial
            ofSetColor(255);
            tutorial.draw(30, 0, ofGetWidth()-60, ofGetHeight()/10*9);
            button(ofGetWidth()/2-largeButtonWidth/2, 620, largeButtonWidth,
largeButtonHeight, "Return");
            break;
        case 2: // Calibration
            ofSetColor(255);
            calibration.draw(1, 1);
            button(ofGetWidth()/2-largeButtonWidth-3, 620, largeButtonWidth,
largeButtonHeight, "Return");
            button(ofGetWidth()/2+3, 620, largeButtonWidth, largeButtonHeight,
"Continue");

            break;
        case 3: // Workout overview
            for (int i = filesToShow ; i <filesToShow+8; i++){
                if (i >= filenames.size()) break;
                button(ofGetWidth()/2-largeButtonWidth/2, 50+ (largeButtonHeight+5)*(i-
filesToShow), largeButtonWidth, largeButtonHeight, filenames[i]);
```

```
            }
            if (filesToShow > 0) button(ofGetWidth()/2+largeButtonWidth/2+10, 50,
smallButtonWidth, largeButtonHeight, "up");
            if (filenames.size()>filesToShow +8)button(ofGetWidth()/2+largeButtonWidth/
2+10, 505, smallButtonWidth, largeButtonHeight, "down");
            button(ofGetWidth()/2-largeButtonWidth/2, 620, largeButtonWidth,
largeButtonHeight, "Return");
            break;
        case 4: // Show workout
            glEnable(GL_DEPTH_TEST);
            display(rotateX, rotateY, newWorkout);
            glDisable(GL_DEPTH_TEST);

            ofNoFill();
            ofSetColor(0);
            ofDrawRectangle(10, ofGetHeight()-90, 20, 80);
            ofFill();
            ofDrawBitmapString("Power", 5,ofGetHeight()-95);
            ofDrawBitmapString("High", 35,ofGetHeight()-83);
            ofDrawBitmapString("Low", 35,ofGetHeight()-7);
            gradient.draw();

            //Stop workout
            button(ofGetWidth()/2-largeButtonWidth/2, 620, largeButtonWidth,
largeButtonHeight, "Stop Workout");


            if(getPauze()){ //pause button
                button(ofGetWidth()/2-smallButtonWidth/2, 550, smallButtonWidth,
smallButtonHeight, " ");
                ofSetColor(0, 0, 0);
                ofNoFill();
                ofDrawTriangle(ofGetWidth()/2-smallButtonWidth/5, 535+smallButtonHeight/
2, ofGetWidth()/2-smallButtonWidth/5, 565+smallButtonHeight/2, ofGetWidth()/
2+smallButtonWidth/4, 550+smallButtonHeight/2);
                ofFill();
            } else { //play button
                button(ofGetWidth()/2-smallButtonWidth/2, 550, smallButtonWidth,
smallButtonHeight, " ");
                ofSetColor(0, 0, 0);
                ofNoFill();
                ofDrawRectangle(ofGetWidth()/2-smallButtonWidth/5-2,
550+smallButtonHeight/4, smallButtonWidth/5, smallButtonHeight/2);
                ofDrawRectangle(ofGetWidth()/2+2, 550+smallButtonHeight/4,
smallButtonWidth/5, smallButtonHeight/2);
                ofFill();
            }

            //Display the time
            int showTime = getToDisplay();
            std::string displayTime = "";
            displayTime.append(std::to_string((showTime-(showTime%3000))/3000));
            displayTime.append(":");
            displayTime.append(std::to_string((showTime%3000)/50));
            ofDrawBitmapString(displayTime, ofGetWidth()/2-11, 540);

            //Go backwards
            button(ofGetWidth()/2-largeButtonWidth/2-smallButtonWidth -10, 550,
smallButtonWidth, smallButtonHeight, " ");
            ofSetColor(0, 0, 0);
            ofNoFill();
            ofDrawTriangle(ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*4,
                           540+smallButtonHeight/2,
                           ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*4,
                           560+smallButtonHeight/2,
                           ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*2.5f,
                           550+smallButtonHeight/2);
            ofDrawTriangle(ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*2.3f,
```

```
                              540+smallButtonHeight/2,
                              ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*2.3f,
                              560+smallButtonHeight/2,
                              ofGetWidth()/2-largeButtonWidth/2-
smallButtonWidth-10+smallButtonWidth/5*0.8f,
                              550+smallButtonHeight/2);
            ofFill();

            //slow backward
            button(ofGetWidth()/2-largeButtonWidth/2, 550, smallButtonWidth,
smallButtonHeight, " ");
            ofSetColor(0, 0, 0);
            ofNoFill();
            ofDrawTriangle(ofGetWidth()/2-largeButtonWidth/2+smallButtonWidth/5*2.3f,
                           540+smallButtonHeight/2,
                           ofGetWidth()/2-largeButtonWidth/2+smallButtonWidth/5*2.3f,
                           560+smallButtonHeight/2,
                           ofGetWidth()/2-largeButtonWidth/2+smallButtonWidth/5*0.8f,
                           550+smallButtonHeight/2);
            ofDrawRectangle(ofGetWidth()/2-largeButtonWidth/2+smallButtonWidth/5*2.5f,
540+smallButtonHeight/2, smallButtonWidth/5*1.2f, 20);
            ofFill();

            //slow forward
            button(ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth, 550,
smallButtonWidth, smallButtonHeight, " ");
            ofSetColor(0, 0, 0);
            ofNoFill();
            ofDrawRectangle(ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth/5*3.7f,
540+smallButtonHeight/2, smallButtonWidth/5*1.2f, 20);
            ofDrawTriangle(ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth/5*2.3f,
                           540+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth/5*2.3f,
                           560+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth/5*1.0f,
                           550+smallButtonHeight/2);
            ofFill();

            //Go forward
            button(ofGetWidth()/2+largeButtonWidth/2+10, 550, smallButtonWidth,
smallButtonHeight, " ");
            ofSetColor(0, 0, 0);
            ofNoFill();
            ofDrawTriangle(ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5+10,
                           540+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5+10,
                           560+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5*2.5f+10,
                           550+smallButtonHeight/2);
            ofDrawTriangle(ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5*2.7f+10,
                           540+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5*2.7f+10,
                           560+smallButtonHeight/2,
                           ofGetWidth()/2+largeButtonWidth/2+smallButtonWidth/5*4.2f+10,
                           550+smallButtonHeight/2);
            ofFill();

            //Add favorites
            button(ofGetWidth()-smallButtonWidth-10, 10, smallButtonWidth,
smallButtonHeight, " ");
            ofSetColor(255);
            addstar.draw(ofGetWidth()-smallButtonWidth-8, 12, smallButtonWidth-4,
smallButtonHeight-4);

            //Show favorites
            if (showFavorites){
                //closing butten
                button(ofGetWidth()-smallButtonWidth-10, ofGetHeight()-
smallButtonHeight-11, smallButtonWidth, smallButtonHeight, "close");

                //Show favorites
```

```
                for (int i = favoritesToShow ; i <favoritesToShow+8; i++){
                    if (i >= favorites.size()) break;
                    std::string favoritename = "";
                    favoritename.append(std::to_string((favorites[i]-(favorites[i]
%3000))/3000));
                    favoritename.append(":");
                    favoritename.append(std::to_string((favorites[i]%3000)/50));
                    button(ofGetWidth()-smallButtonWidth-10, ofGetHeight()-
smallButtonHeight-10- (smallButtonHeight+5)*(i-favoritesToShow +1), smallButtonWidth,
smallButtonHeight, favoritename);
                    ofSetColor(255);
                    cross.draw(ofGetWidth()-20, ofGetHeight()-smallButtonHeight-10-
(smallButtonHeight+5)*(i-favoritesToShow +1), 10, 10);
                }

                //add scrolling capability
                if (favoritesToShow > 0) button(ofGetWidth()-(smallButtonWidth+10)*1.5f,
ofGetHeight()-2*smallButtonHeight-15, smallButtonWidth/2, smallButtonHeight, "down");
                if (favorites.size()>favoritesToShow +8) button(ofGetWidth()-
(smallButtonWidth+10)*1.5f, ofGetHeight()-9*smallButtonHeight-50, smallButtonWidth/2,
smallButtonHeight, "up");
            } else { // show open menu for favorites
                button(ofGetWidth()-smallButtonWidth-10, ofGetHeight()-
smallButtonHeight-10, smallButtonWidth, smallButtonHeight, " ");
                ofSetColor(255);
                star.draw(ofGetWidth()-smallButtonWidth-8, ofGetHeight()-
smallButtonHeight-8, smallButtonWidth-4, smallButtonHeight-4);
            }

            if(getEof())button(ofGetWidth()/2-100, 100, 200, 60, "End of savefile");

            if (sureToExit){
                button(ofGetWidth()/2-245, 200, 490, 200, "Are you sure to Exit?");
                button(ofGetWidth()/2-235, 395-largeButtonHeight, 150, largeButtonHeight,
"Cancel");
                if (newWorkout)button(ofGetWidth()/2-75, 395-largeButtonHeight, 150,
largeButtonHeight, "Review workout");
                button(ofGetWidth()/2+85, 395-largeButtonHeight, 150, largeButtonHeight,
"Main menu");
            }

            break;
    }
}

//The control for looking around
int xOld;
int yOld;
bool first = true;
//--------------------------------------------------------------
void ofApp::mouseDragged(int x, int y, int button){
    if(first){ //the first time always compes the dragged value with the previous, which
can result in big differences if the mouse has been moved
        xOld = x;
        yOld = y;
        first = false;
    }
    rotateX+=x;
    rotateX-=xOld;
    rotateY-=y;
    rotateY+=yOld;
    xOld=x;
    yOld=y;
}

//--------------------------------------------------------------
void ofApp::mousePressed(int x, int y, int button){
    switch (state){
        case 0: // Main menu
            if (buttonClick(ofGetWidth()/2-largeButtonWidth/2, 100, largeButtonWidth,
largeButtonHeight, x, y)) state = 1;
```

```
            if (buttonClick(ofGetWidth()/2-largeButtonWidth/2, 200, largeButtonWidth,
largeButtonHeight, x, y)) state = 2;
            if (buttonClick(ofGetWidth()/2-largeButtonWidth/2, 300, largeButtonWidth,
largeButtonHeight, x, y)) {
                state = 3;
                readFolder(&filenames);
            }
            break;
        case 1: // Tutorial
            if( buttonClick(ofGetWidth()/2-largeButtonWidth/2, 600, largeButtonWidth,
largeButtonHeight, x, y)) state = 0; //return
            break;
        case 2: // Calibration
            if( buttonClick(ofGetWidth()/2-largeButtonWidth-3, 600, largeButtonWidth,
largeButtonHeight, x, y)) state = 0; //return
            if( buttonClick(ofGetWidth()/2+3, 600, largeButtonWidth, largeButtonHeight,
x, y)) { //continue
                state = 4;
                newWorkout = true;
                rotateX = 0;
                rotateY = 0;
                setForce(getForce());
            }
            break;
        case 3: // Workout overview

            for (int i = filesToShow ; i <filesToShow+8; i++){ //load workout
                if (i >= filenames.size()) break;
                if (buttonClick(ofGetWidth()/2-largeButtonWidth/2, 50+
(largeButtonHeight+5)*(i-filesToShow), largeButtonWidth, largeButtonHeight, x, y)){
                    state = 4;
                    rotateX = 0;
                    rotateY = 0;
                    readFile(filenames[i]);
                    break;
                }
            }
            if (filesToShow > 0 && buttonClick(ofGetWidth()/2+largeButtonWidth/2+10, 50,
smallButtonWidth, largeButtonHeight, x, y)) filesToShow-=8;
            if (filenames.size()>filesToShow +8 && buttonClick(ofGetWidth()/
2+largeButtonWidth/2+10, 505, smallButtonWidth, largeButtonHeight, x, y)) filesToShow+=8;

            if( buttonClick(ofGetWidth()/2-largeButtonWidth/2, 600, largeButtonWidth,
largeButtonHeight, x, y)) state = 0; //return
            break;
        case 4: // Show workout
            if( buttonClick(ofGetWidth()/2-largeButtonWidth/2, 620, largeButtonWidth,
largeButtonHeight, x, y)) { //stop workout
                sureToExit = true;
                setPauze(true);
            }

            //The notification that shows up when you stop a workout
            if(sureToExit){


                if (buttonClick(ofGetWidth()/2+85, 395-largeButtonHeight, 150,
largeButtonHeight, x, y)){ //Main menu
                    state = 0;
                    if (newWorkout){
                        closefile();
                    } else {
                        addFavoritesTofile();
                    }
                    endDisplay();
                    favoritesToShow = 0;
                    showFavorites = false;
                    newWorkout = false;
                    sureToExit = false;
                } else if (newWorkout && buttonClick(ofGetWidth()/2-75, 395-
largeButtonHeight, 150, largeButtonHeight, x, y)){ //Review workout
                    newWorkout = false;
```

```
                    setToDisplay(0);
                    sureToExit = false;
                    setPauze(false);
                } else if (buttonClick(ofGetWidth()/2-235, 395-largeButtonHeight, 150,
largeButtonHeight, x, y)) { //Cancel
                    sureToExit = false;
                    setPauze(false);
                }

            }

            //pauze button
            if (buttonClick(ofGetWidth()/2-smallButtonWidth/2, 550, smallButtonWidth,
smallButtonHeight, x, y)) setPauze(!getPauze());
            if (buttonClick(ofGetWidth()/2-largeButtonWidth/2-smallButtonWidth -10, 550,
smallButtonWidth, smallButtonHeight, x, y)) changeSpeed(0, -2.0f); //Fast backwards
            if (buttonClick(ofGetWidth()/2-largeButtonWidth/2, 550, smallButtonWidth,
smallButtonHeight, x, y)) changeSpeed(0, -0.5f); //Slow backward
            if (buttonClick(ofGetWidth()/2+largeButtonWidth/2-smallButtonWidth, 550,
smallButtonWidth, smallButtonHeight, x, y)) changeSpeed(0, 0.5f); //Slow forward
            if (buttonClick(ofGetWidth()/2+largeButtonWidth/2+10, 550, smallButtonWidth,
smallButtonHeight, x, y)) changeSpeed(0, 2.0f); //fast forward

            //Add favorite
            if( buttonClick(ofGetWidth()-smallButtonWidth-10, 10, smallButtonWidth,
smallButtonHeight, x, y)) favorites.push_back(getToDisplay());

            //Open or close the favorites menu
            if (buttonClick(ofGetWidth()-smallButtonWidth-10, ofGetHeight()-
smallButtonHeight-11, smallButtonWidth, smallButtonHeight, x, y)) showFavorites = !
showFavorites;

            //If the menu is opened the favorite buttons become selectable
            if(showFavorites){
                for (int i = favoritesToShow ; i <favoritesToShow+8; i++){
                    if (i >= favorites.size()) break;

                    //Erase favorites
                    if (buttonClick(ofGetWidth()-20, ofGetHeight()-smallButtonHeight-10-
(smallButtonHeight+5)*(i-favoritesToShow +1), 10, 10, x, y)) {
                        favorites.erase(favorites.begin() + i);
                        break;
                    }

                    //Select favorite
                    if (buttonClick(ofGetWidth()-smallButtonWidth-10, ofGetHeight()-
smallButtonHeight-11- (smallButtonHeight+5)*(i-favoritesToShow +1), smallButtonWidth,
smallButtonHeight, x, y)){
                        setToDisplay(favorites[i]);
                        setPauze(true);
                        break;
                    }
                }
                //Up and down buttons, which are only visible if you can go up or down
                if (favoritesToShow > 0 && buttonClick(ofGetWidth()-
(smallButtonWidth+10)*1.5f, ofGetHeight()-2*smallButtonHeight-15, smallButtonWidth/2,
smallButtonHeight, x, y))favoritesToShow-=8;
                if (favorites.size()>favoritesToShow +8 && buttonClick(ofGetWidth()-
(smallButtonWidth+10)*1.5f, ofGetHeight()-9*smallButtonHeight-50, smallButtonWidth/2,
smallButtonHeight, x, y))favoritesToShow+=8;
            }

            break;
    }

}

//--------------------------------------------------------------
void ofApp::mouseReleased(int x, int y, int button){
    first = true; //The next time the mouse will be dragged the first-statement will run
again
}
```

Collector.cpp

```
#ifndef collector_cpp
#define collector_cpp

// Copyright (C) 2013-2014 Thalmic Labs Inc.
// Distributed under the Myo SDK license agreement. See LICENSE.txt for details.
#define _USE_MATH_DEFINES
#include <cmath>
#include <iostream>
#include <iomanip>
#include <stdexcept>
#include <string>
#include <algorithm>

// The only file that needs to be included to use the Myo C++ SDK is myo.hpp.
#include <myo/myo.hpp>
#include <ofmain.h>

// Classes that inherit from myo::DeviceListener can be used to receive events from Myo
devices. DeviceListener
// provides several virtual functions for handling different kinds of events. If you do
not override an event, the
// default behavior is to do nothing.
class DataCollector : public myo::DeviceListener {
public:
    DataCollector()
    : onArm(false), isUnlocked(false)
    {
    }

    // These values store the EMG values
    int mav;
    int emgData[8];//TO-DO check whether 8 is the wanted amount

    //rotation stores the to-be-viewed rotation
    myo::Vector3<float> rotation;

    //making timestamp a global variable so it can be stored in a file in readdata
    uint64_t timedata;

    // onUnpair() is called whenever the Myo is disconnected from Myo Connect by the
user.
    void onUnpair(myo::Myo* myo, uint64_t timestamp)
    {
        // We've lost a Myo.
        // Let's clean up some leftover state.
        onArm = false;
        isUnlocked = false;
    }

    void onConnect( myo::Myo*    myo, uint64_t timestamp, myo::FirmwareVersion
firmwareVersion){
        myo->setStreamEmg(myo::Myo::streamEmgEnabled);

    }

    // onOrientationData() is called whenever the Myo device provides its current
orientation, which is represented
    // as a unit quaternion.
    void onOrientationData(myo::Myo* myo, uint64_t timestamp, const
myo::Quaternion<float>& quat)
    {
        myo::Quaternion<float> rotateQuat(-0.5f,0.5f,0.5f,0.5f);
        rotateQuat = rotateQuat*quat;
        rotateQuat = rotateQuat.normalized();

        rotation = myo::Vector3<float>(250,0,0);
        rotation = rotate(rotateQuat, rotation);

        if (mav == 0) myo->setStreamEmg(myo::Myo::streamEmgEnabled);
        timedata=timestamp;
```

```
    }

    //onEmgData() is called whenever the Myo provides its current emg data
    void onEmgData(myo::Myo* myo, uint64_t timestamp, const int8_t* emg){
        int length = sizeof(emgData)/sizeof(emgData[0]);

        // Make room for new data by moving all data entries up by 1
        for(size_t j = 0; j <= length-2; j++){
            emgData[j]=emgData[j+1];
        }

        //insert the new values in new EmgData
        emgData[length-1]=0;
        for (size_t i = 0; i < 8; i++) {
            emgData[length-1]+=abs(static_cast<int>(emg[i]));
        }

        //Moving average filter
        mav=0;

        //Add all stored emg values together
        for(size_t j = 0; j < length-1; j++){
            mav+=emgData[j];
        }

        //divide by amount of values
        mav/=length;
    }

    // onArmSync() is called whenever Myo has recognized a Sync Gesture after someone has
put it on their
    // arm. This lets Myo know which arm it's on and which way it's facing.
    void onArmSync(myo::Myo* myo, uint64_t timestamp, myo::Arm arm, myo::XDirection
xDirection, float rotation,
                   myo::WarmupState warmupState)
    {
        onArm = true;
        whichArm = arm;
    }

    // onArmUnsync() is called whenever Myo has detected that it was moved from a stable
position on a person's arm after
    // it recognized the arm. Typically this happens when someone takes Myo off of their
arm, but it can also happen
    // when Myo is moved around on the arm.
    void onArmUnsync(myo::Myo* myo, uint64_t timestamp)
    {
        onArm = false;
    }

    // onUnlock() is called whenever Myo has become unlocked, and will start delivering
pose events.
    void onUnlock(myo::Myo* myo, uint64_t timestamp)
    {
        isUnlocked = true;
    }

    // onLock() is called whenever Myo has become locked. No pose events will be sent
until the Myo is unlocked again.
    void onLock(myo::Myo* myo, uint64_t timestamp)
    {
        isUnlocked = false;
    }

    // These values are set by onArmSync() and onArmUnsync() above.
    bool onArm;
    myo::Arm whichArm;

    // This is set by onUnlocked() and onLocked() above.
    bool isUnlocked;
};
#endif /* collector_cpp */
```

readdata.cpp

```
#include "ofmain.h"
#include "readdata.hpp"
#include "display.hpp"
#include <fstream>
#include <boost/filesystem.hpp>

vector<dataStorage> data;
vector<int> favorites;
bool newFile;
int emgMax;
ofstream savefile;

//Inserts data into the Myo
void insertData(DataCollector* collector){
    dataStorage newData;
    newData.timestamp = collector->timedata;
    newData.force = collector->mav;
    newData.rotation = ofVec3f (collector->rotation.x(), collector-
>rotation.y() ,collector->rotation.z());
    data.push_back(newData);
    if (newFile){
        string filename = "savefiles/final";
        filename.append(std::to_string(collector->timedata));
        filename.append(".txt");
        savefile.open(filename, ios::trunc | ios::out);
        newFile = false;
        savefile << emgMax << '\n';
        emgMax = 0;
    }
    savefile << newData.timestamp << "," << newData.force << "," << newData.rotation.x <<
"," << newData.rotation.y << "," << newData.rotation.z <<'\n';
}

void closefile(){
    if( favorites.size() > 0){
        savefile << "f";
        for (int i = 0; i < favorites.size() ;i++){
            savefile << "," << favorites[i];
        }
        savefile << '\n';

    }
    savefile.close();
}

//Get the maximum force during the calibration
void calibrate(DataCollector* collector){
    if (emgMax < collector->mav) emgMax = collector->mav;
}

//Allows the measured force to be used as calibration. The emgMax gets cleared so it can
be used next time
int getForce(){
    newFile = true;
    return emgMax;
}

//Reads all the files in the folder
void readFolder(vector<std::string>* filenames){
    filenames->clear();
    boost::filesystem::path p("savefiles");
    for (auto i = boost::filesystem::directory_iterator(p); i !=
boost::filesystem::directory_iterator(); i++)
    {
        if (!boost::filesystem::is_directory(i->path())) //we eliminate directories in a
list
        {
            filenames->push_back(i->path().filename().string());
        }
        else
            continue;
```

```
    }
}

//Reads the data from the Myo
void readFile(std::string filename){
    std::string location = "savefiles/";
    location.append(filename);
    ifstream datafile;
    datafile.open(location , ios::in);

    std::string istring;

    getline(datafile, istring);
    int force = std::stoi(istring);
    setForce(force);

    while (!datafile.eof()){
        dataStorage newData;
        getline(datafile, istring, ',');

        if (istring == "f" || datafile.eof()) break;

        std::istringstream iss(istring);
        iss >> newData.timestamp;
        getline(datafile, istring, ',');
        newData.force = std::stoi(istring);
        getline(datafile, istring, ',');
        newData.rotation.x = std::stof(istring);
        getline(datafile, istring, ',');
        newData.rotation.y = std::stof(istring);
        getline(datafile, istring);
        newData.rotation.z = std::stof(istring);

        data.push_back(newData);
    }
    if (istring== "f"){
        while (!datafile.eof()){
            getline(datafile, istring, ',');
            favorites.push_back(std::stoi(istring));
        }
    }

    datafile.close();
}

//Adds the favorites to the file by completely rewriting the file
void addFavoritesTofile(){
    string filename = "savefiles/workout";
    filename.append(std::to_string(data[0].timestamp));
    filename.append(".txt");
    ofstream datafile;
    datafile.open(filename, ios::out |ios::trunc);

    datafile << getForceMax() << '\n';

    for (int i = 0; i < data.size(); i++){
    datafile << data[i].timestamp << "," << data[i].force << "," << data[i].rotation.x <<
"," << data[i].rotation.y << "," << data[i].rotation.z <<'\n';
    }

    if( favorites.size() > 0){
        datafile << "f";
        for (int i = 0; i < favorites.size() ;i++){
            datafile  << "," << favorites[i];
        }
        datafile << '\n';

    }
    datafile.close();
}
```

display.cpp

```
#include "display.hpp"
#include "readdata.hpp"
#include "ofMain.h"
#include "extraFunctions.hpp"

float toDisplay = 0;
float forceMax = 0;
float displaySpeed = 1.0f;
bool pauze = false, eof = false;

void display(int rotateY, int rotateX, bool newWorkout){
    //Draw the course of the swing

    ofSetColor(100,200,100);
    ofPushMatrix();
    ofTranslate(300, 275);
    ofRotateX(rotateX/2);
    ofRotateY(rotateY/2);
    ofPushMatrix();
    ofTranslate(0, 700);
    ofRotateX(90);
    //ofRotateY(rotateY/2);
    //ofTranslate(0, 225);
    ofDrawCircle(0, 0, 100000);
    ofPopMatrix();
    ofPopMatrix();


    ofPushMatrix();
    ofTranslate(300, 275);
    ofRotateX(rotateX/2);
    ofRotateY(rotateY/2);
    int i = toDisplay-1;
    while(i > toDisplay-50 && i > 0 && toDisplay<data.size()) {
        int mav = data[i].force;
        ofVec3f rotation = data[i].rotation;
        ofVec3f rotationOld = data[i-1].rotation;
        if (forceMax < mav) forceMax = mav;
        ofColor color;
        if (mav<forceMax/2){
            color.b=255-(mav/forceMax*255);
            color.g=mav/forceMax*510;
            color.r=0;
        } else {
            color.b=0;
            color.g=510-(mav/forceMax*510);
            color.r=mav/forceMax*255;
        }

        //Set the color based upon the power of the muscles
        ofSetColor(color, 255-(toDisplay-i)*4);

        //The swing consists of various rectangles, connecting the current value with the
previous one, which creates the visualisation of a swing
        ofBeginShape();
        ofVertex(rotation.x/2, rotation.y/2, rotation.z/2);
        ofVertex(rotation.x, rotation.y, rotation.z);
        ofVertex(rotationOld.x, rotationOld.y, rotationOld.z);
        ofVertex(rotationOld.x/2, rotationOld.y/2, rotationOld.z/2);
        ofEndShape();
        i--;
    }

    //These are the axis shown
    ofSetColor(0, 0, 0);
    ofDrawLine(-250,0,0,    250,0,0);
    ofDrawLine(0,250,0,    0,0,0);
    ofDrawLine(0,0,-250,    0,0,250);
    ofSetColor(150,0,0);
    ofDrawLine(0,0,0,    0,-250,0);
    ofPopMatrix();
```

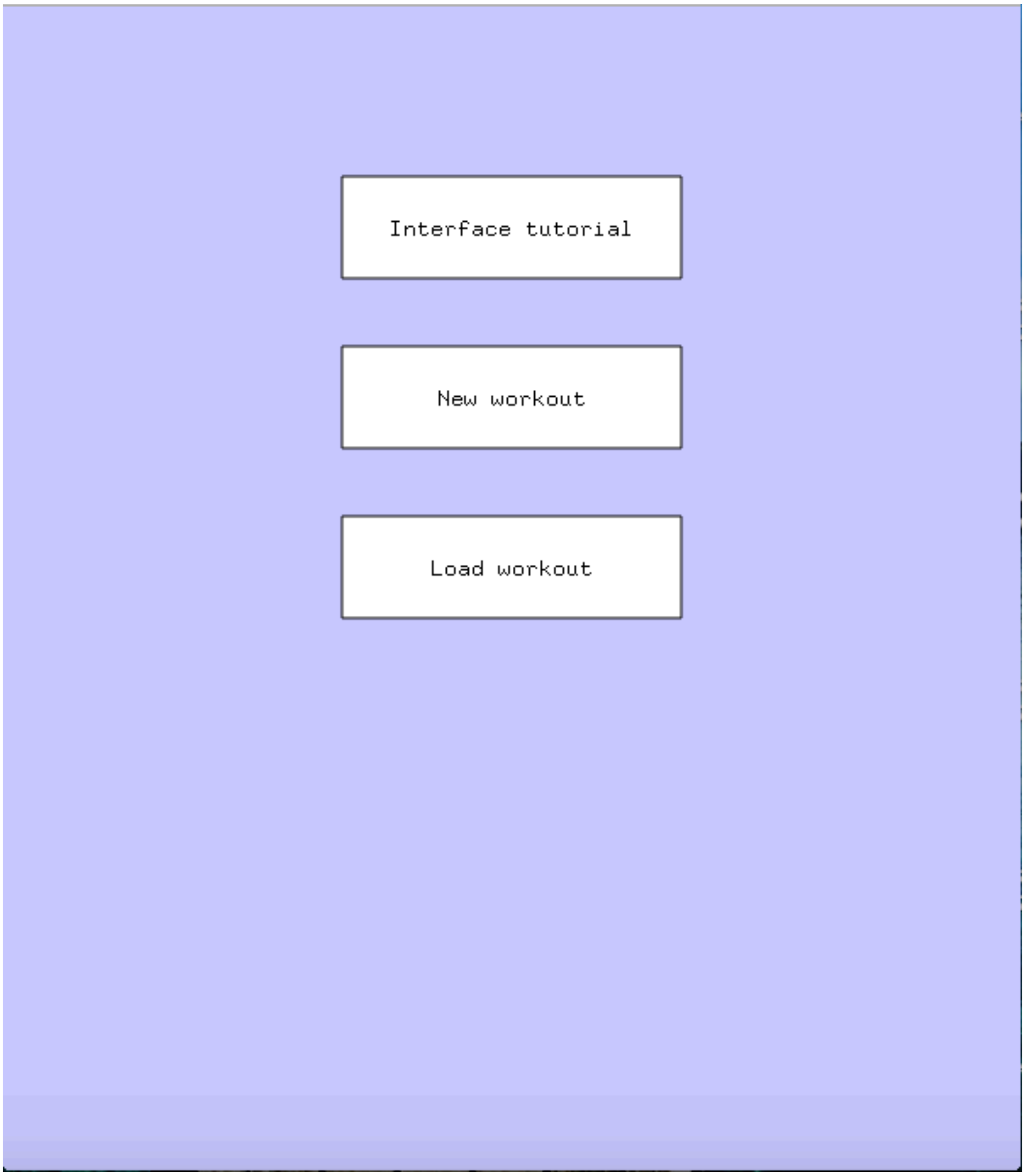
```
    if (!pauze) toDisplay+=displaySpeed;
    if (toDisplay>=data.size() && displaySpeed > 1.0f && newWorkout) {
        toDisplay = data.size()-1;
        displaySpeed = 1.0f;
    } else if (toDisplay>=data.size()-1 && displaySpeed >= 0.0f && !newWorkout) {
        toDisplay = data.size()-1;
        displaySpeed = 0.0f;
        eof = true;
    } else {
        eof = false;
    }
    if (toDisplay < 0 && displaySpeed < 0) {
        toDisplay = 0;
        displaySpeed = 0;
        pauze = true;
    }

}

//clear up leftover state once the workout is finished
void endDisplay(){
    toDisplay = 0;
    data.clear();
    favorites.clear();
    forceMax = 0;
    displaySpeed = 1.0f;
    pauze = false;
}

void changeSpeed(int setSpeed, float timesSpeed){
    if (timesSpeed < 0 && displaySpeed >= 0){ //if moving forward and selecting to go
backward, set the speed
        displaySpeed= timesSpeed;
    } else if ((timesSpeed < 0 && displaySpeed < 0)){ //if moving backward and selecting
to go backward, change by how much
        displaySpeed= -abs(displaySpeed*timesSpeed);
    } else if (timesSpeed > 0 &&  displaySpeed <= 0){ //if moving backward and selecting
to go forward, set the speed
        displaySpeed= timesSpeed;
    } else if (timesSpeed > 0 &&  displaySpeed > 0){ //if moving forward and selecting to
go forward, change by how much
        displaySpeed*= timesSpeed;
    }
    if (setSpeed!=0){
        displaySpeed = setSpeed;
    }
    pauze = false;
}

bool getPauze(){
    return pauze;
}

void setPauze(bool toSet){
    pauze = toSet;
    displaySpeed = 1.0f;
}

float getForceMax(){
    return forceMax;
}

void setForce(int calibratedForce){
    forceMax = calibratedForce;
}

float getToDisplay(){
    return toDisplay;
}

void setToDisplay(int favorite){
```

```
    toDisplay = favorite;
}

bool getEof(){
    return eof;
}
```

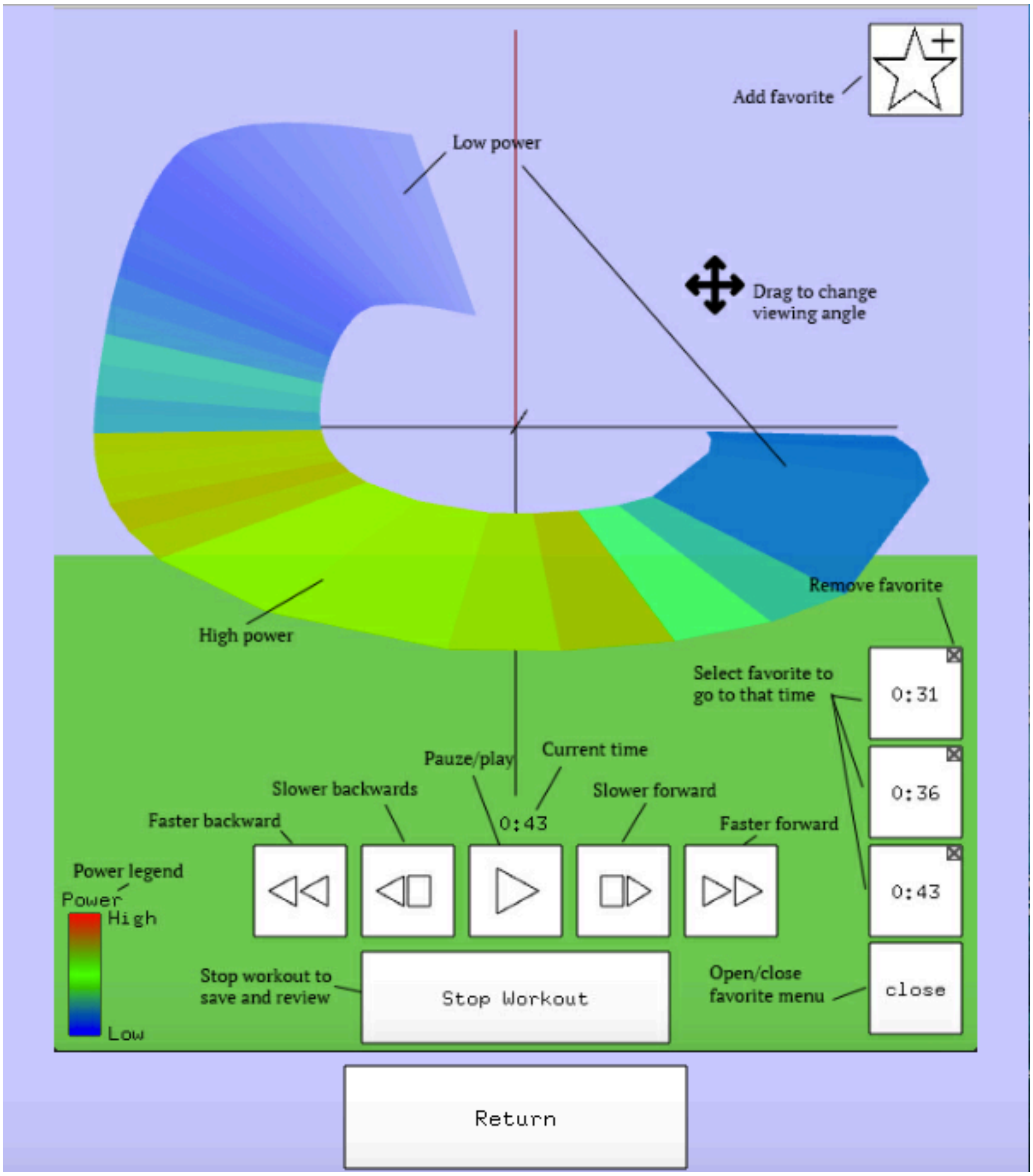
extraFunctions.cpp

```
#include "extraFunctions.hpp"
#include "ofMain.h"

// This function creates a button
void button(int x, int y, int width, int height, std::string text){
    ofSetColor(255);
    ofDrawRectangle(x, y, width, height);
    ofNoFill();
    ofSetColor(0);
    ofDrawRectangle(x, y, width, height);
    ofFill();
    int length = text.size()*8;
    ofSetColor(0, 0, 0);
    ofDrawBitmapString(text, x+(width/2-length/2),y+height/2+5);
}

//This function checks whether a button has been clicked
bool buttonClick(int x, int y, int width, int height, int mouseX, int mouseY){
    bool clicked = false;
    if (mouseX > x && mouseX < x+width && mouseY > y && mouseY < y+height) clicked =
true;
    return clicked;
}
```

Appendix H: Visualisations of Hi-Fi prototype

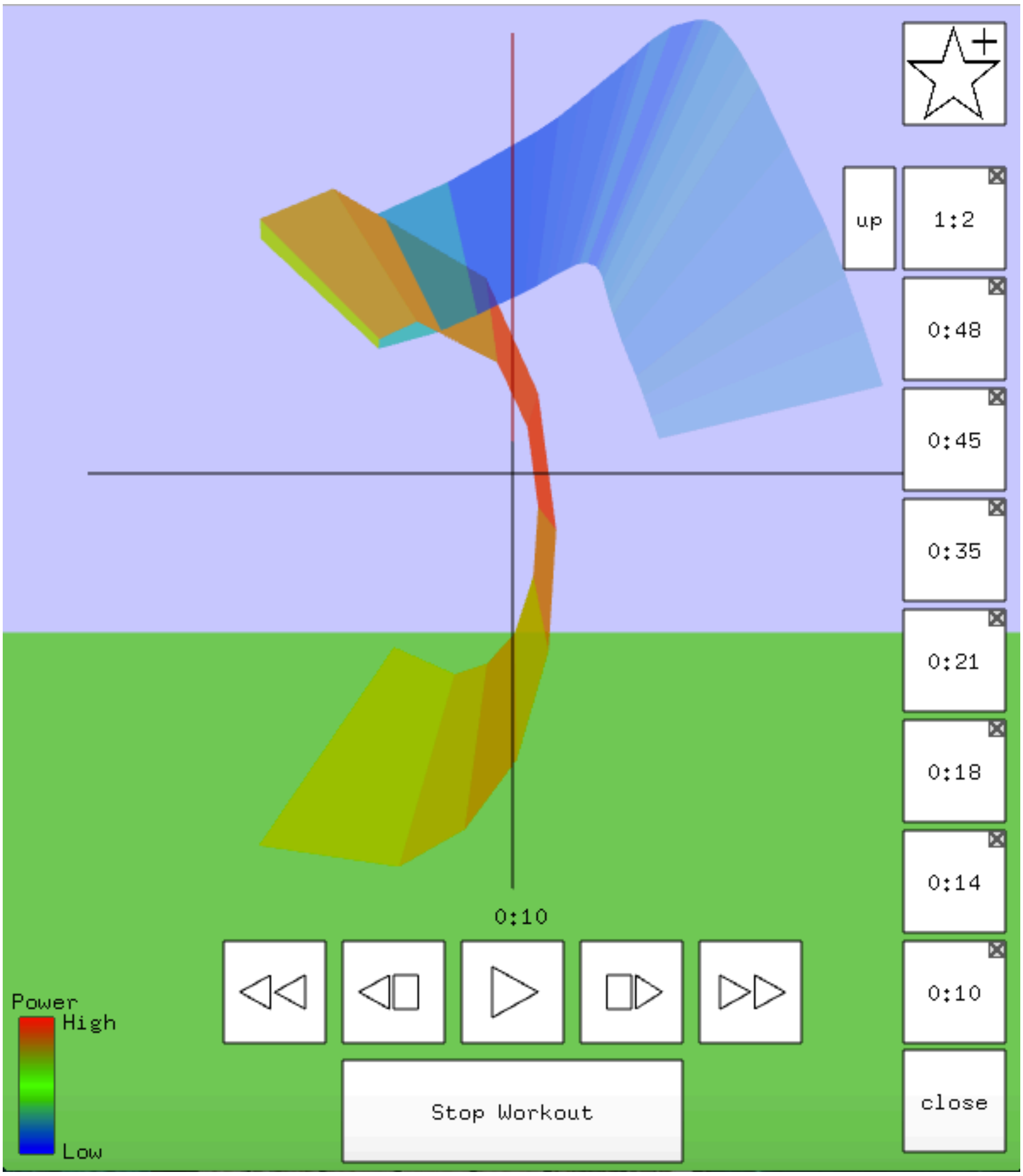


Visualisation of the main menu

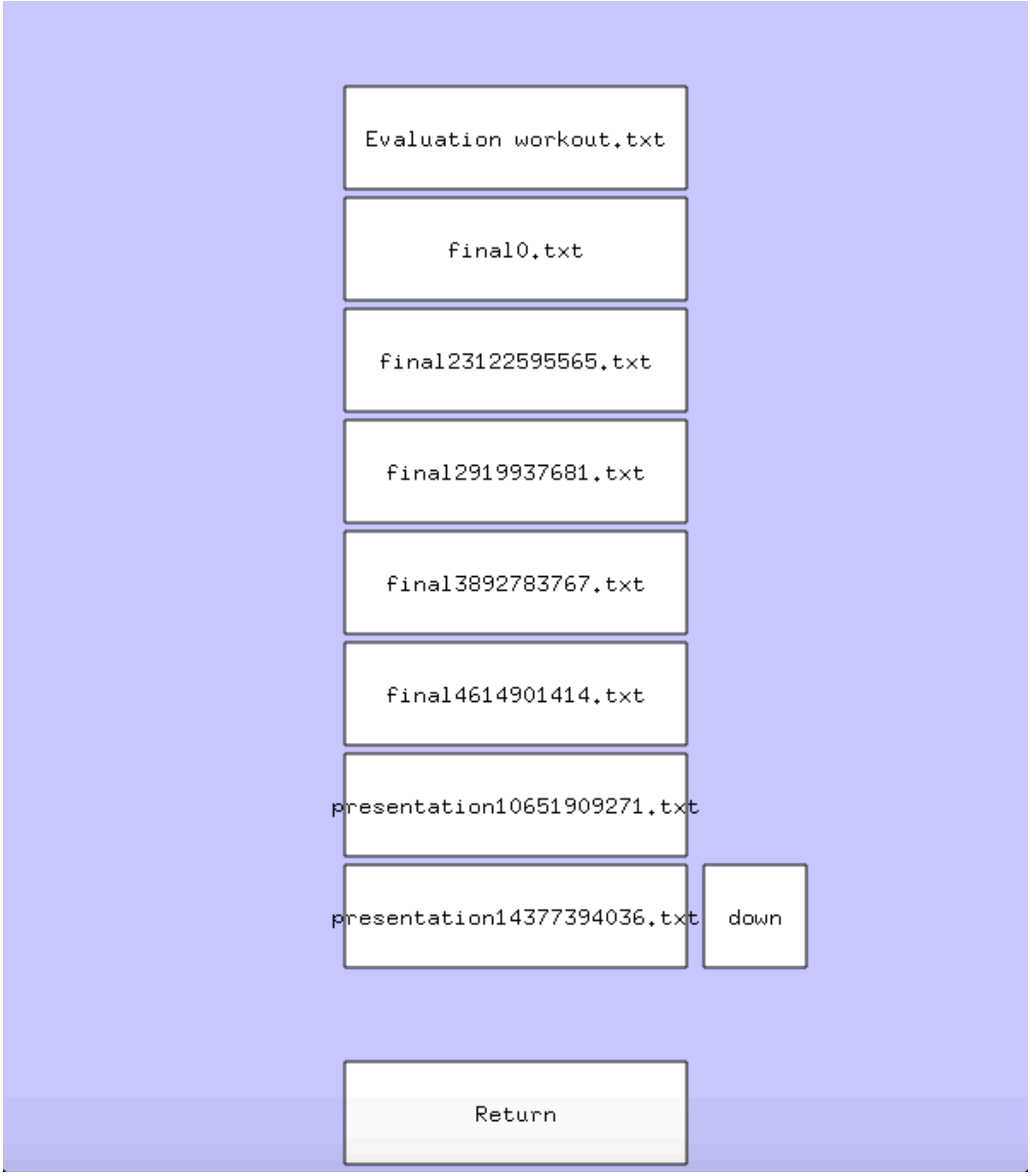


Visualisation of the interface tutorial

Visualisation of the calibration



Visualisation of the workout



Visualisation of the load workout menu

Appendix I: Consent form

Title of research: Myo Tennis Trainer

Researcher: Kevin Vogelzang

To be filled in by subject

I hereby declare that I have been informed in a manner which is clear to me about the nature and method of this research. My questions have been answered to my satisfaction. I agree of my own free will to participate in this research. I reserve the right to withdraw this consent without the need to give any reason and I am aware that I may withdraw from the experiment at any time.

If my research results are to be used in scientific publications or made public in any other manner, then they will be made completely anonymous. My personal data will not be disclosed to third parties without my express permission. If I request further information about the research, now or in the future, I may contact Kevin Vogelzang; email: k.h.vogelzang@student.utwente.nl

If you have any complaints about this research, please direct them to the secretary of the Ethics Committee of the Faculty of Electrical Engineering, Mathematics and Computer Science at the University of Twente, dr. ir. J.F.C. Verberne, P.O. Box 217, 7500 AE Enschede (NL), telephone: +31 (0)53 489 3700; email: j.f.c.verberne@utwente.nl).

Name subject

…………………………………………………

Signature Subject

…………………………………………………

Date: ……………………

To be filled in by researcher.

I have provided explanatory notes about the research. I declare myself willing to answer to the best of my ability any questions which may still arise about the research.'

Name researcher

…………………………………………………

Signature Researcher

…………………………………………………

Date: ……………………

Evaluation assignments

Thank you for participating in this evaluation. This evaluation will probably take around 10 minutes. This sheet contains a number of assignments for you to complete. For each assignment there is first a short use scenario, afterwards the assignment is given.

Assignment 1

During this assignment you are going to play a quick game of tennis using the provided application.

- Go to the “Interface tutorial” in order to get a quick overview of the interface.
- Once done start a new workout
- (Pretend to) Play tennis (the interviewer will tell you when to stop)
- “Pause” the workout
- From now on the Myo is not needed anymore, if it is uncomfortable you can take it off.
- Please fill in survey 1.

Assignment 2

After the game you want to look how well you did by reviewing your workout.

- Go to “Review workout”
- Look at the third swing.
  - Is it a backhand, forehand or serve? (keep in mind for the survey or tell the interviewer)
- Save it as a “favourite”
- Check whether your favourite has been saved in the menu.
- Go back to the “main menu”
- Please fill in survey 2.

Assignment 3

You are now together with your trainer and want to show a swing you did earlier and ask for feedback.

- Open the workout “”Evaluation Workout.txt”
- The interviewer will now ask you to look at certain swings and identify them. To answer these questions correctly, you must know that the user in this scenario is right-handed.
- Please fill in survey 3.

Thank you for participating in this evaluation!

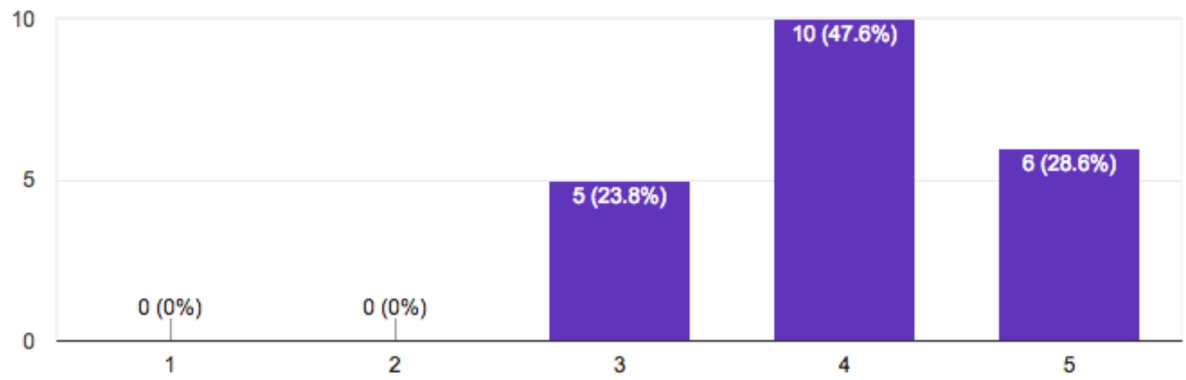
Appendix K: User evaluation results

Below the results of the survey are displayed according to the requirements and further on based upon topic.

FR1: The app must use the Myo

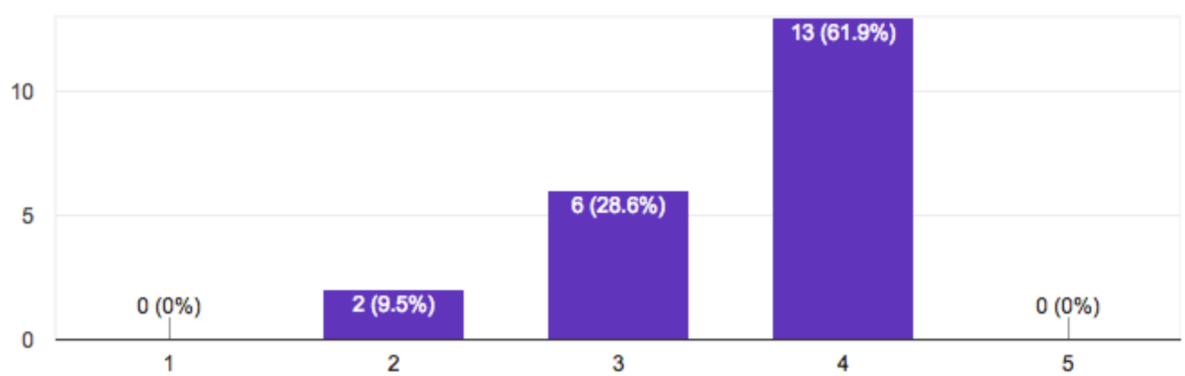
What did you think about wearing the Myo? Uncomfortable - comfortable

21 responses



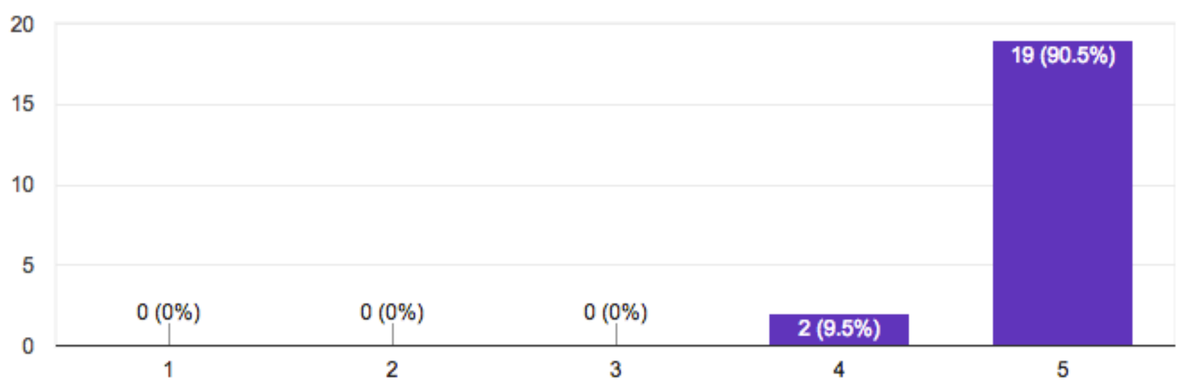
Distracting - unnoticeable

21 responses



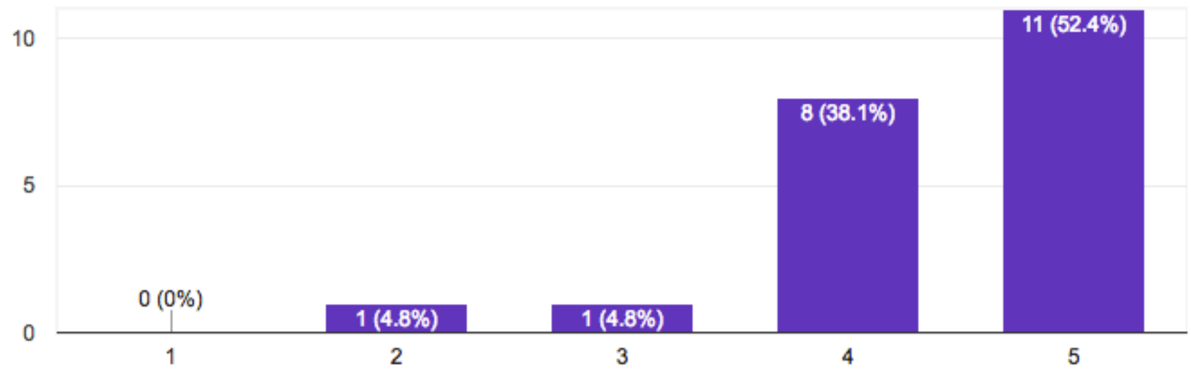
How were your movements while wearing the Myo? Hindered - free

21 responses



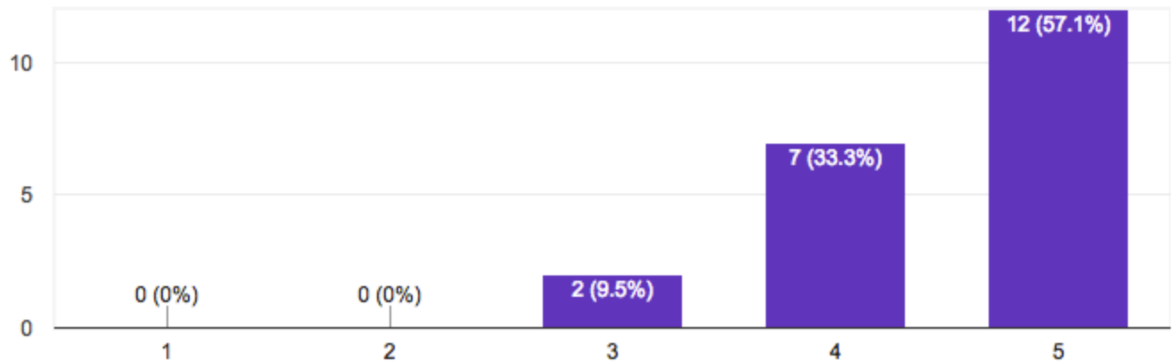
What do you think of using the Myo as a feedback system for tennis? Illogical - logical

21 responses



Impractical - practical

21 responses



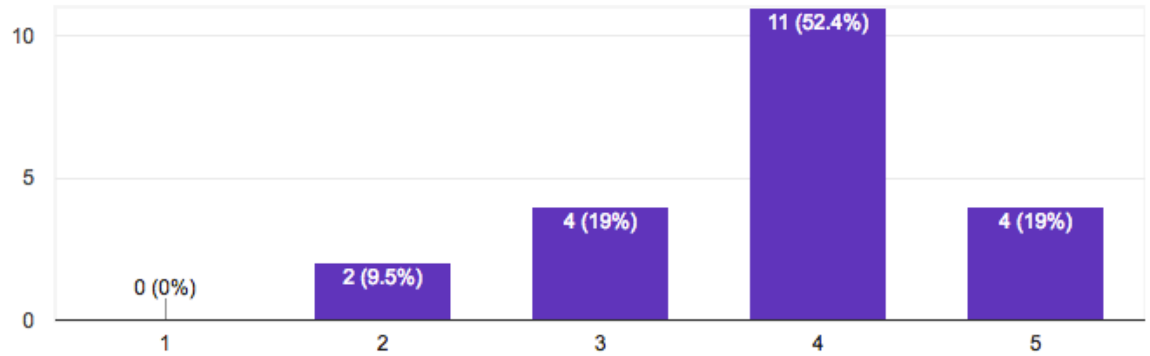
FR2: The app must visualise the movement of the player using the movement data of the Myo

Comments regarding this topic:

- Hard to find out what the relevant beginning and ending of the movement were
- The orientation of the movements were a bit unclear, especially the depth
- An animated hand/arm.
- A slowmotion/visual of myself
- Something recognisable in the interface of the ' movements' . something like a human in the center so you can relate the movements to something
- The tracking system works awesome :)
- Clear front and back, left and right. Automatic swing recognition. When paused, something that implicates the direction of movement.
- Permanent forward marker so I don't lose track of the orientation, slowmo?
- An arrow in what direction the movement was. Additionally an option to go back a set amount of time (3 seconds for example to skip over moves approximately).
- Video images of the swing
- It is hard to recognise how the movement relates to the arm. however the speed and power are easily being recognised.
- The orientation of the player
- Nice visualisations were made
- Interpreting the data does not come instinctive but will probably be clearer when the system is used more often.

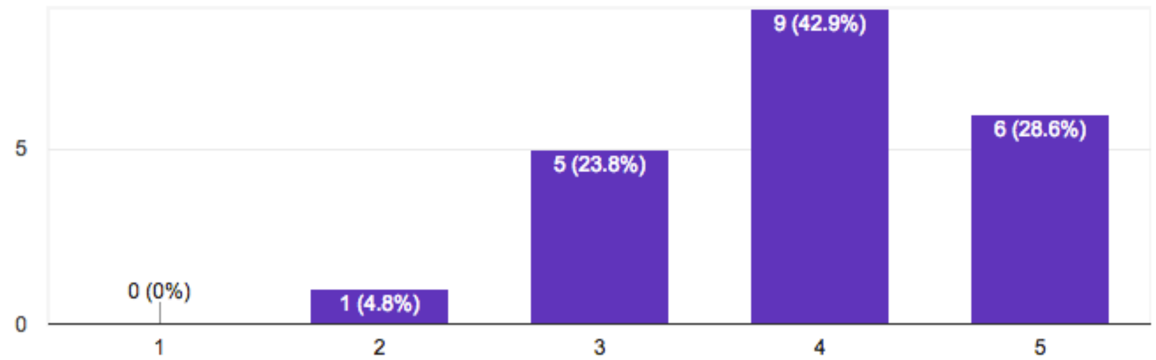
What did you think of your movements? Unfamiliar - recognisable

21 responses



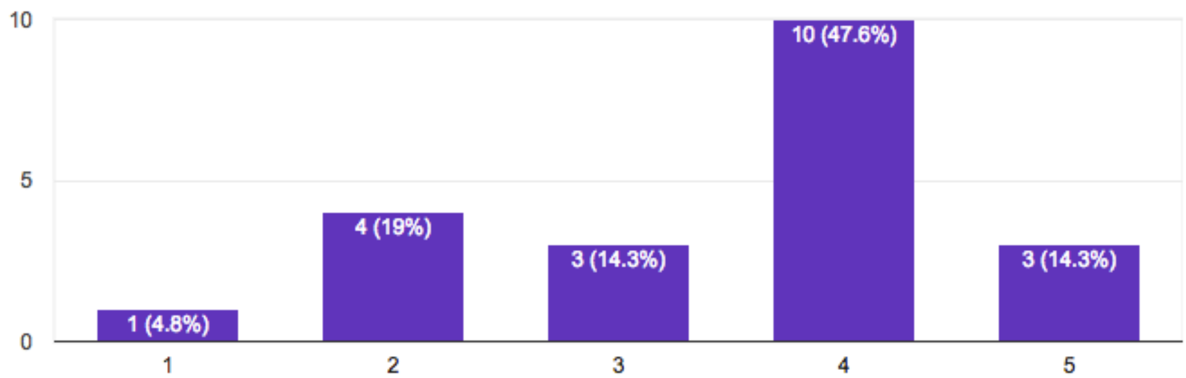
What did you think of the visualisation? Unclear - clear

21 responses



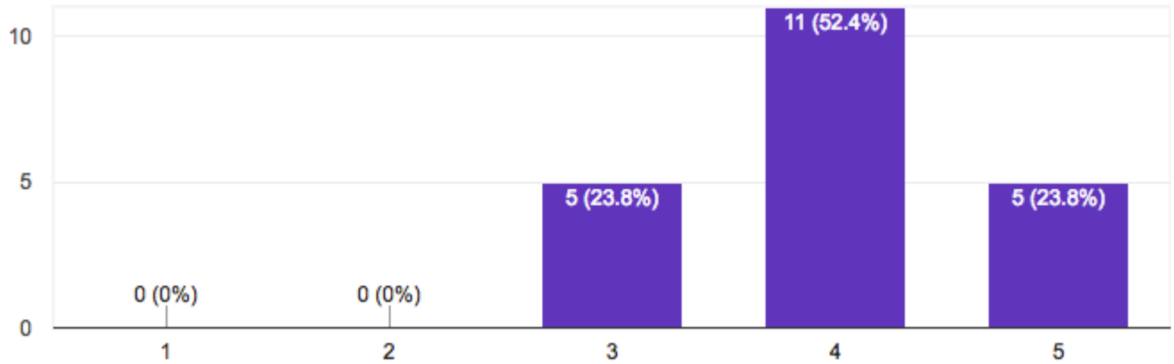
Difficult - easy

21 responses



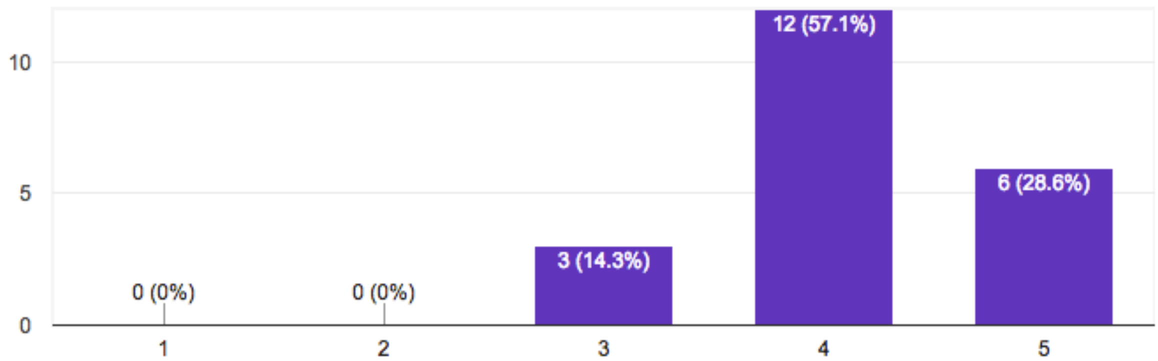
Useless - useful

21 responses



Incomprehensible - understandable

21 responses

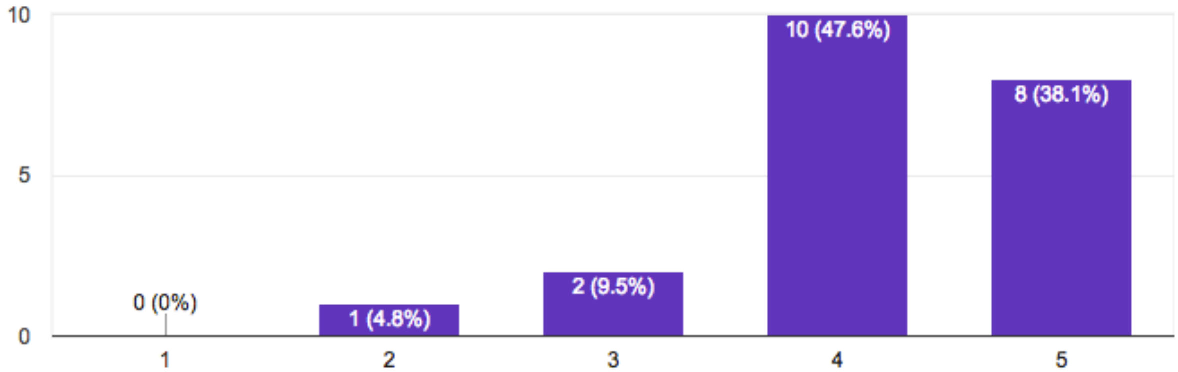


FR3: The app must show the power used by the muscles synchronised with the movement
Comments regarding this topic:

- I did not know what the power actually meant (did I not pay attention?)
- Maybe a change in colors, like less powerful swing green and a powerful swing red
- As said before, the power scale didn't work for me. However if it does show the different scales it can be really useful
- Make it clearer that green is not good, but actually weak. had a little legend at the bottom or something
- Make a clear explanation of power and what the consequences of much/less power are

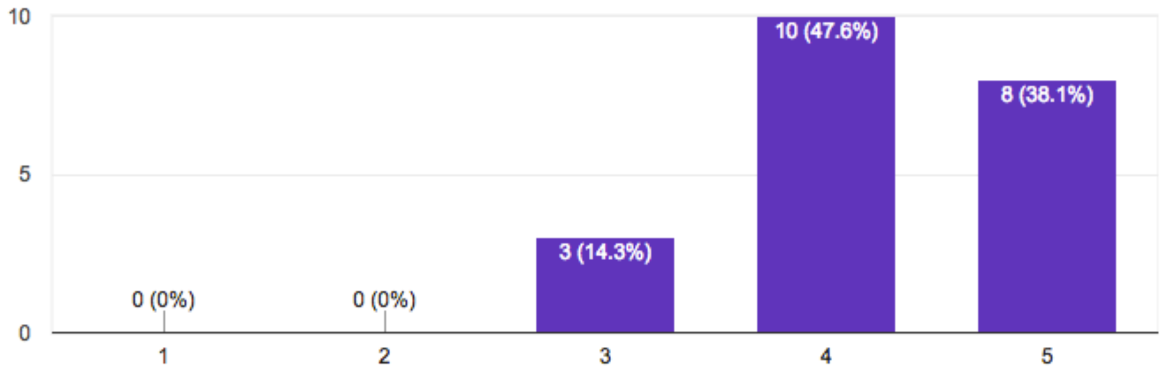
What did you think of the power? Unfamiliar - recognisable

21 responses



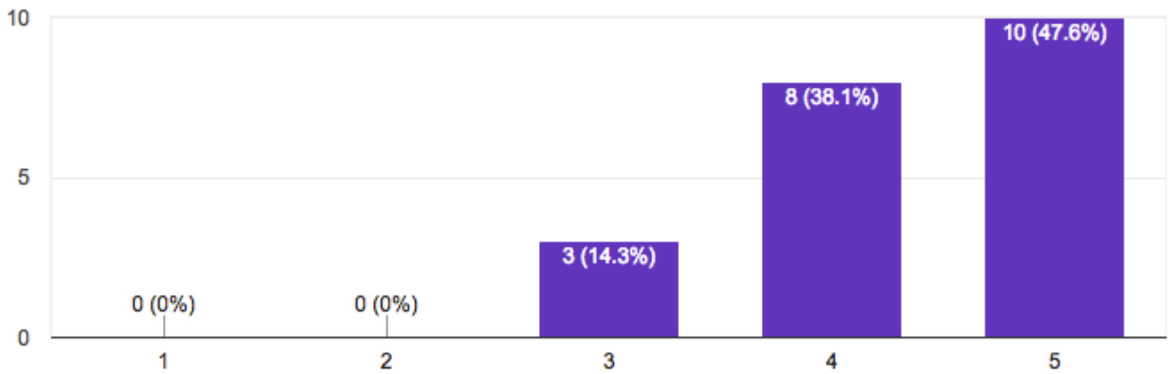
Useless - useful

21 responses



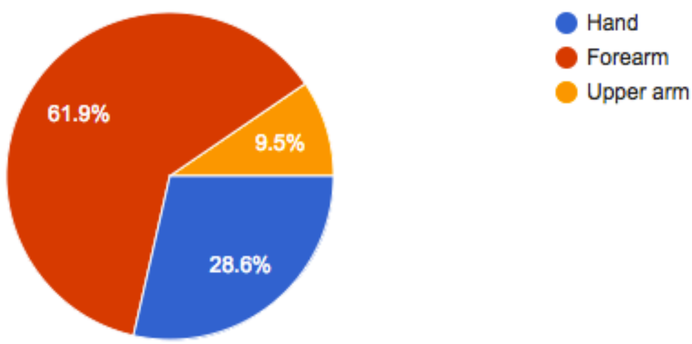
Unclear - clear

21 responses



The power of the muscles controlling what part of your body was visualised?

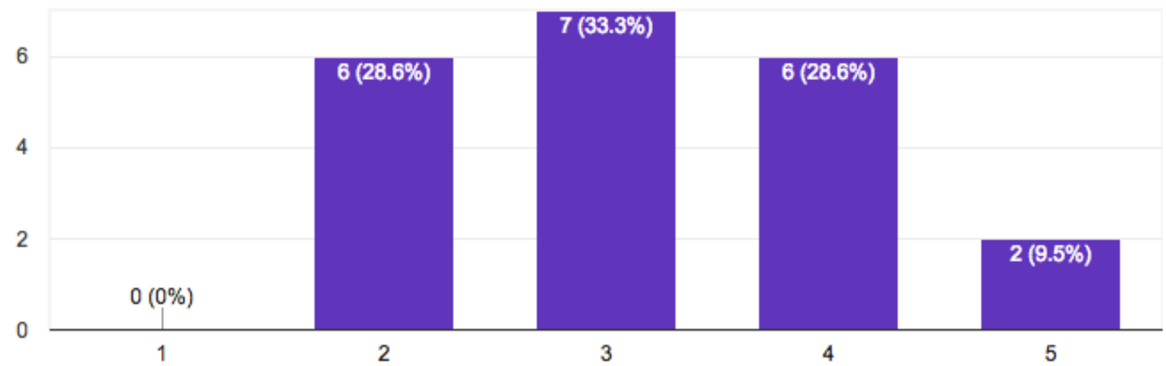
21 responses



FR4: The muscle power and movement data must be calibrated in order to match the player and correctly store the orientation.

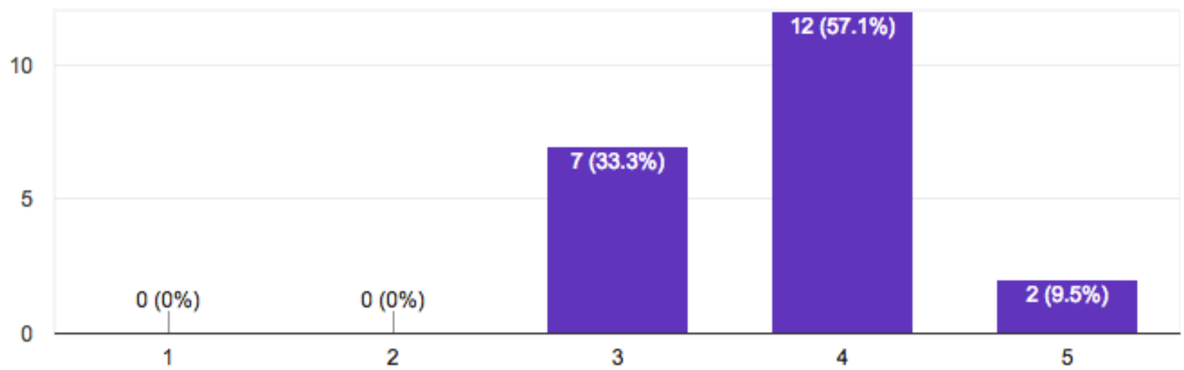
What did you think about the calibration? Unclear - clear

21 responses



Annoying - Pleasant

21 responses



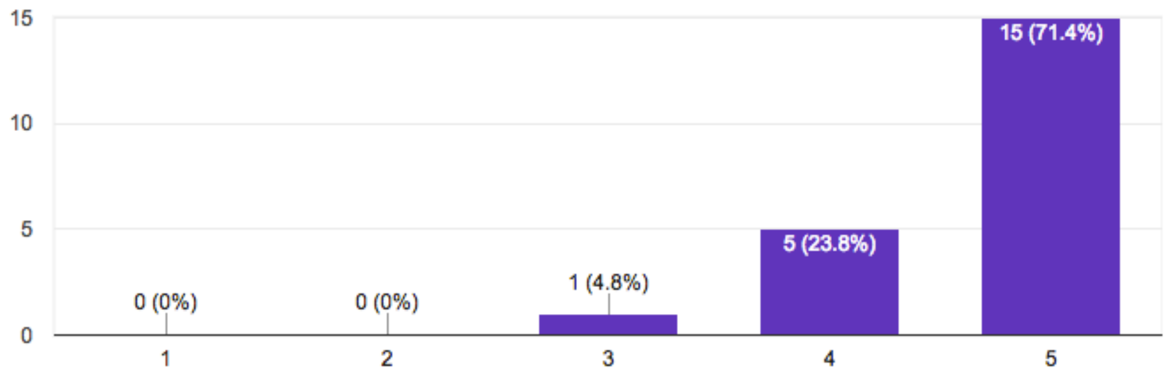
FR5: The app must store full trainings/matches & FR6: The app must be able to show the stored trainings/matches

Comments regarding this topic:

- Naming workouts and favorites
- Comprehensive naming of workouts (use a real date e.g. 2017-06-27, 15:30-16:30).
- The ability to give a workout a specific name

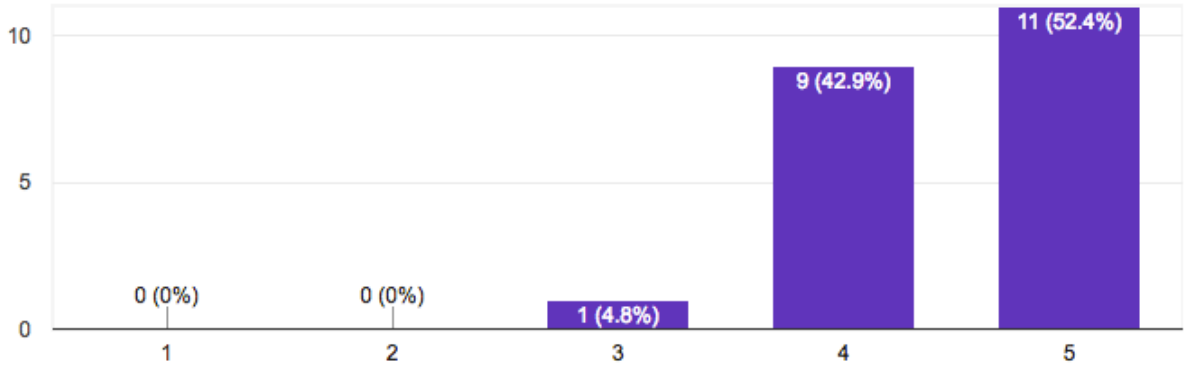
What did you think of selecting an existing workout? Difficult - easy

21 responses



Unclear - clear

21 responses



FR7: The user must have the ability to pause, play, fast forward and go backwards in the recordings.

Comments regarding this topic:

- Make a timeline (similar to a music timeline showing the bass) this represents movement. you can slide over it to view your swings. when making a favorite you can select a beginning and an ending. Would be a lot better than play, stop and reverse forward buttons. (would also allow for faster scrolling) And biggest pro is that you get rid of the ugly buttons.
- A time slider would be nice I think

FR8: The app should show the movement and power synchronised with a video of the player

Comments regarding this topic:

- Video images of the swing
- A slowmotion/visual of myself

FR9: The app should recognise swings and only show the swings, discarding the movement in between

Comments regarding this topic:

- Maybe a sound feedback could be used to indicate the swing movement.
- Hard to find out what the relevant beginning and ending of the movement were
- Let the application directly recognise a backhand/forehand service

FR10: The app should have the feature of storing separate swings for later viewing

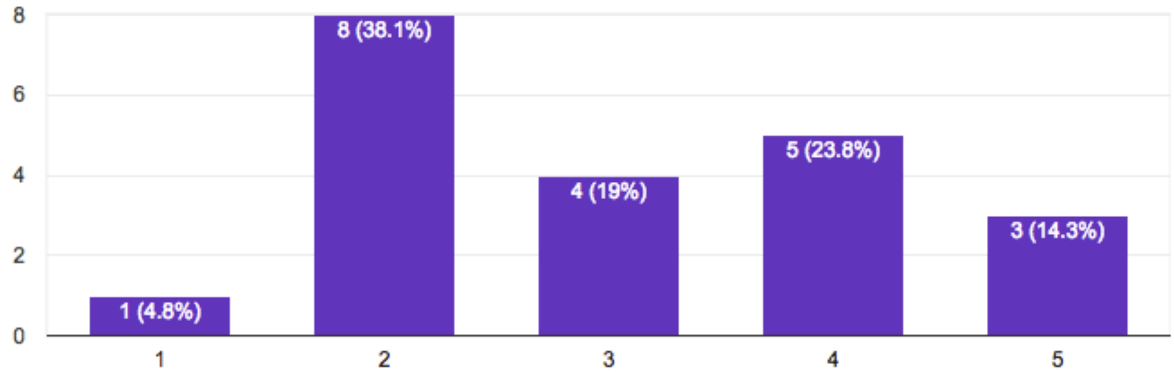
Comments regarding this topic:

- I didn't really understand what I favourited when I pressed the favourite button. At first I thought I favourited the entire workout.
- Adding the same swing to favourites multiple times
- There is no instant feedback while saving a favourite, it would be good to have a small pop up that says favorite is saved. So had I three times the same favourite.
- Took a little time to understand the add favourite feature.
- I did not realise the favourite feature was actually saving the information. Some feedback here would be very nice.
- Feedback when something is added to the favourite menu
- Not clear how to create a favorite
- It was unclear that I added a favourite.
- Favorite refers to a time instead of a move.
- I did not see the favourite button on top, so I was looking at the bottom star how to add the hit to the menu
- No clear feedback when a favourite was saved. The favourites-menu icon could be more clear.
- the feedback regarding add to favorites is not fully intuitive

- It is difficult to exactly pinpoint the time that you want to save. In addition, it was unclear to me that the favourites button had to be clicked to see your saved favourites.
- no feedback when you save a favourite
- I had to figure out how the favourites option exactly works. After that, it was easy to use
- When making a favorite you can select a beginning and an ending.
- Animation for saving the favourite, indication what I actually save (I thought the whole path that was displayed at that time?)
- Maybe add an icon to the favourites button that indicates it can be extended.
- Let the user decide the beginning and ending of a 'favourite' instead of one determined moment

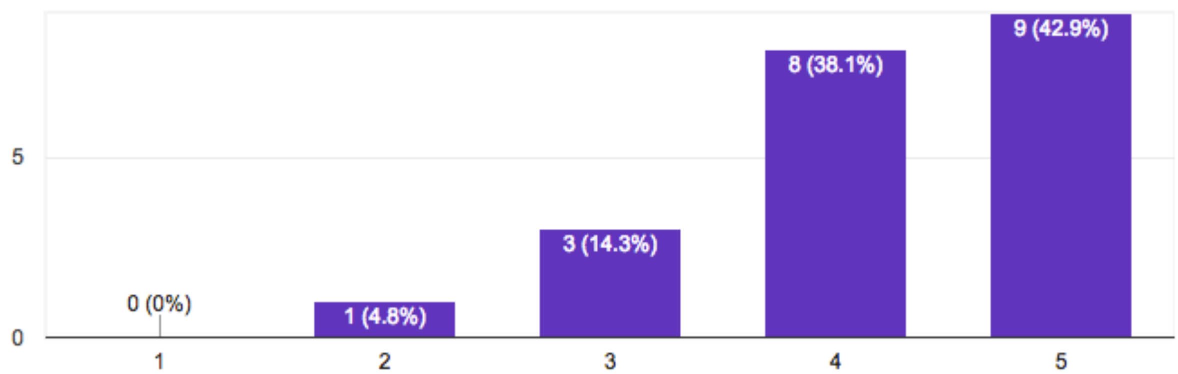
What did you think about the “favourite” feature? Unclear - clear

21 responses



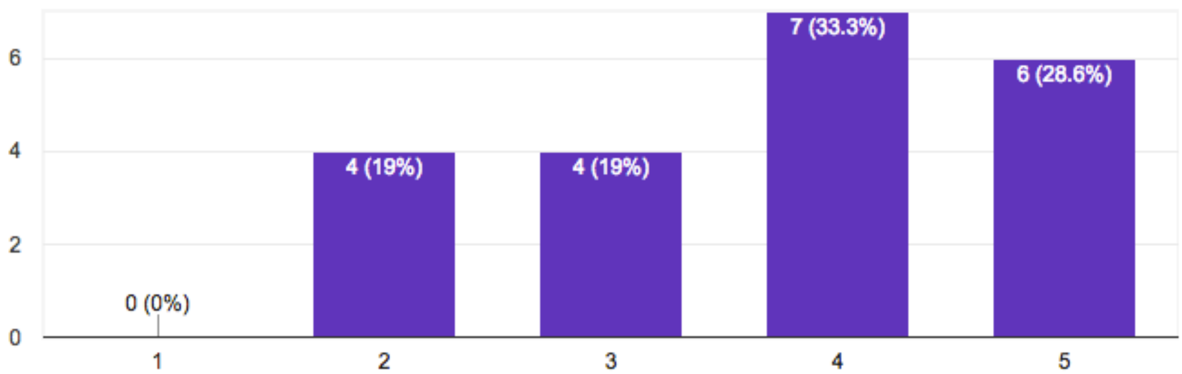
Useless - useful

21 responses

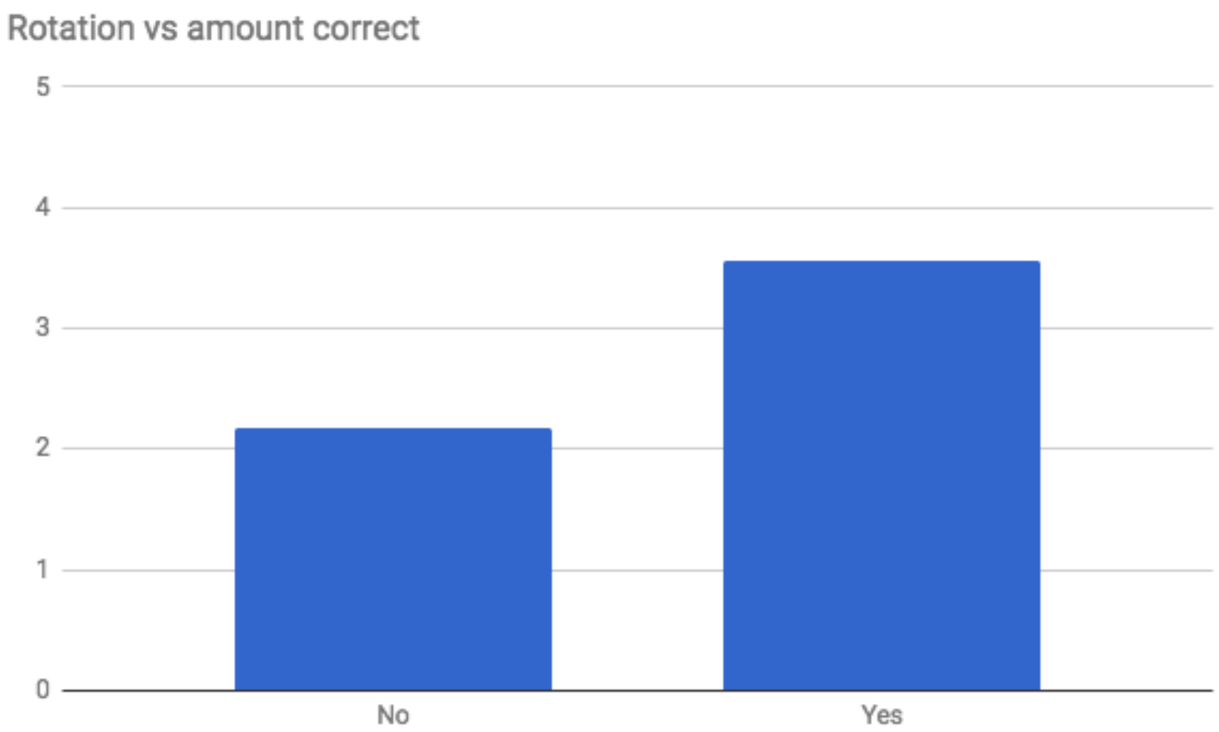


Difficult - easy

21 responses



FR12: The app should be able to change the viewing angle of the swings



FR13: The app could give an overview of various statistics
Comments regarding this topic:
- Maybe that the Myo will identify automatically what a forehand, backhand or serve is? Maybe an indicator what an average is for your age, gender or level? So you can compare yourselves to other players and make a competition out of it.

FR14: The app could give textual and visual feedback on the swings
Comments regarding this topic:
- Qualitative feedback
- More knowledge about what would be the right way to actually "perform" a forehand, backhand or serve (especially for people who don't play tennis (yet))
- It may be good (but also very difficult because everyone tennis differently) to have a perfect swing programmed so people can repeat that one.
- have a 'follow the line' option, just like singstar. or a workout option, would be fun, bit wii-ish
- Examples of how swings "should be"
- It doesn't know the differences between forehand and backhand.
- Knowledge of backhand etc when not familiar with tennis.
- It would be nice to get some feedback from the program about what you want to see here. It was not clear to me if what I was seeing was good or bad.
- An option to compare your own forehand for example to the forehand of someone with a perfect technique. Direct feedback to show if you hit a forehand, backhand or serve so that you now that it works and that it is calibrated the right way
- Feedback from the application about the quality of your swings or what you're seeing on the screen.

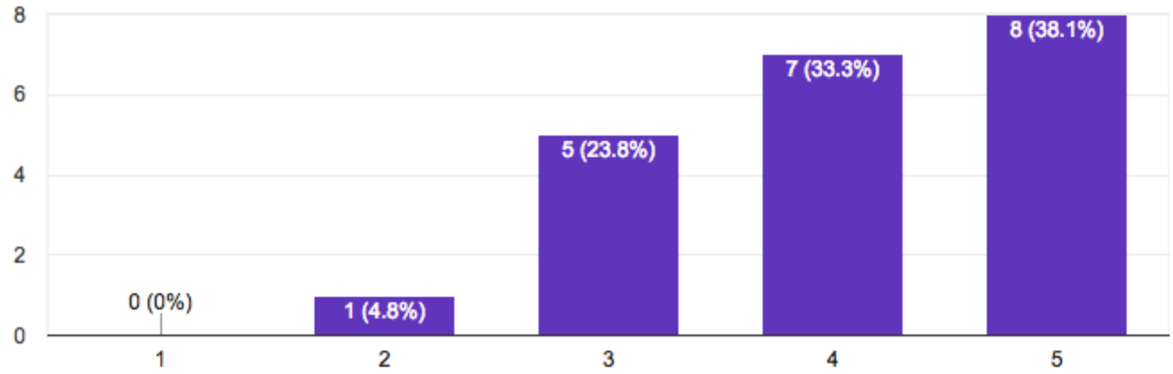
smartphone. This could perhaps be due to the fact that the program was clearly a prototype.

NFR2: The app must have a proper menu allowing the user to easily select what they intend to do
Comments regarding this topic:
- Menu needs a little update, maybe add some more menus instead of one cluttered mess?
- Clear font is used for the menus.
- Fancy design. I find it illogical that 'return to menu' is only visible after you click 'stop workout', same goes for 'review workout’.
- Main menu was hard to find

- Quick button with explanation of menu.

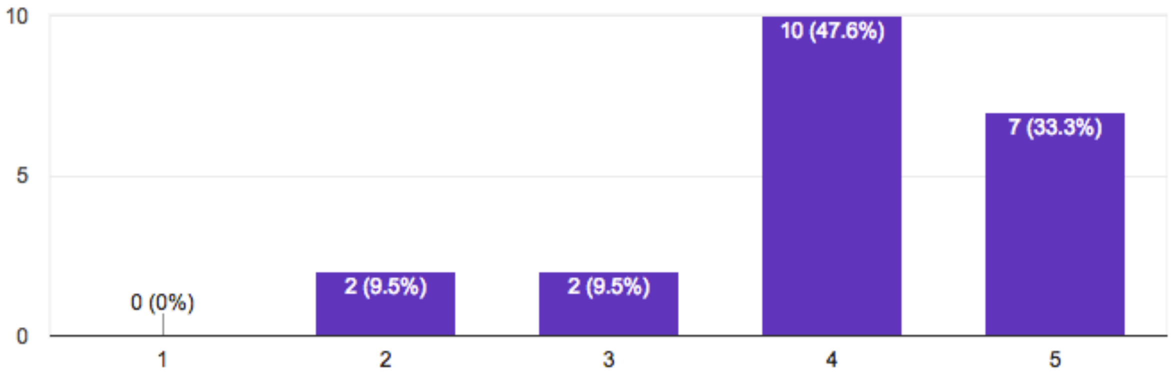
What did you think of the menu? unclear-clear

21 responses



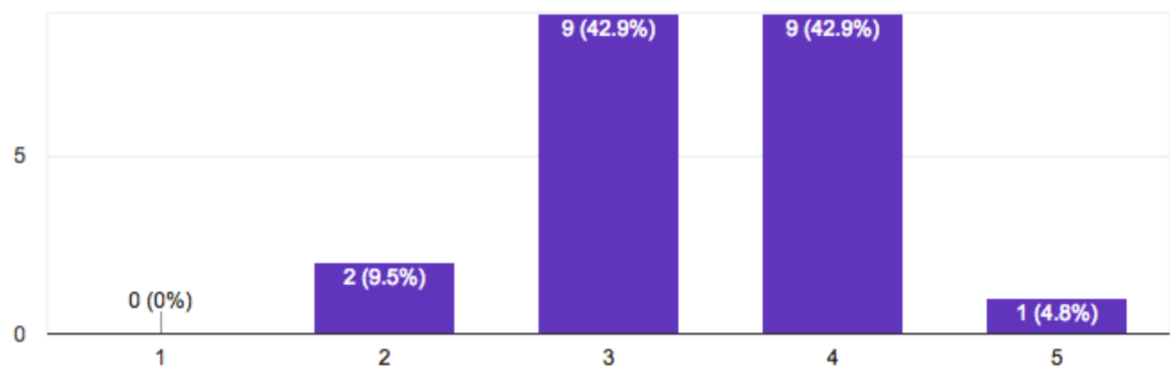
Useless - useful

21 responses



Lacking - extensive

21 responses

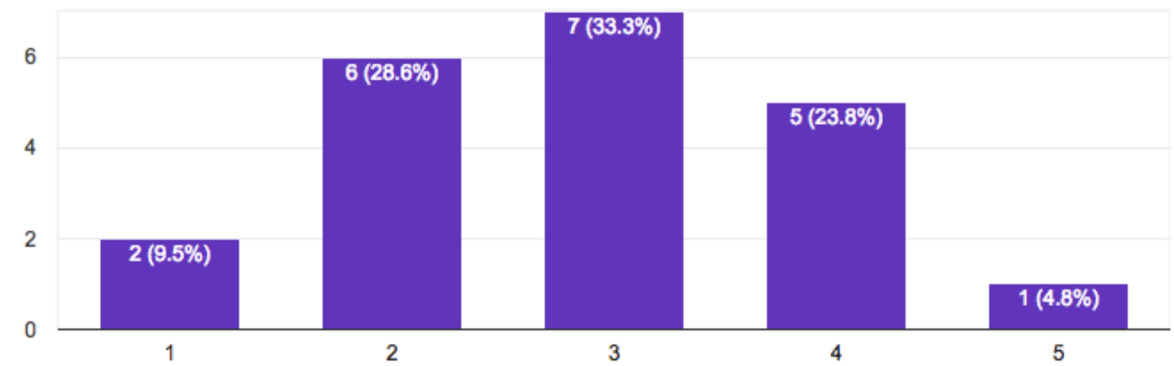


The tutorial
Comments regarding this topic:
- Mostly the tutorial was very vague
- To make the interface more clear, make the most important parts more distinctive
- The interface tutorial is very busy, so it is difficult to precisely see what does what.

- at the interface tutorial it is a bit weird that the legend is left bottom and not instead of the swing motion. makes it confusing in the beginning.
- The font is a bit small, makes it harder to read than necessary

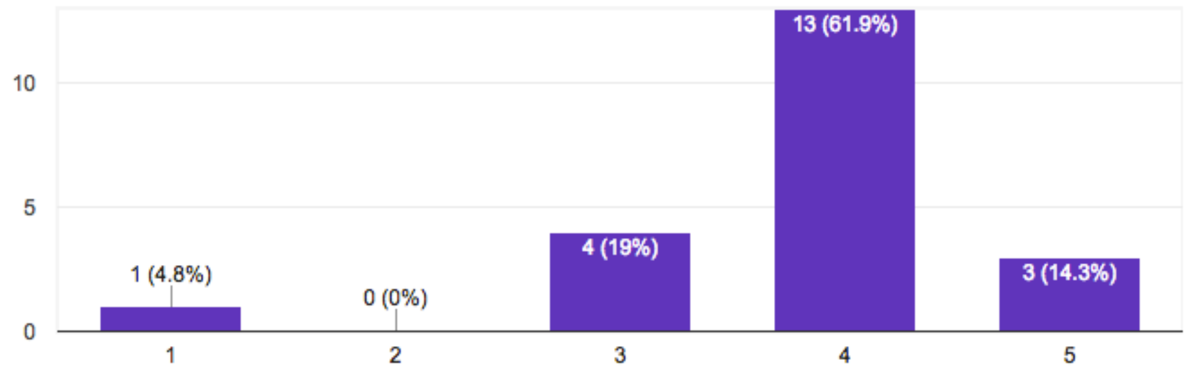
What did you think about the interface tutorial? unclear-clear

21 responses



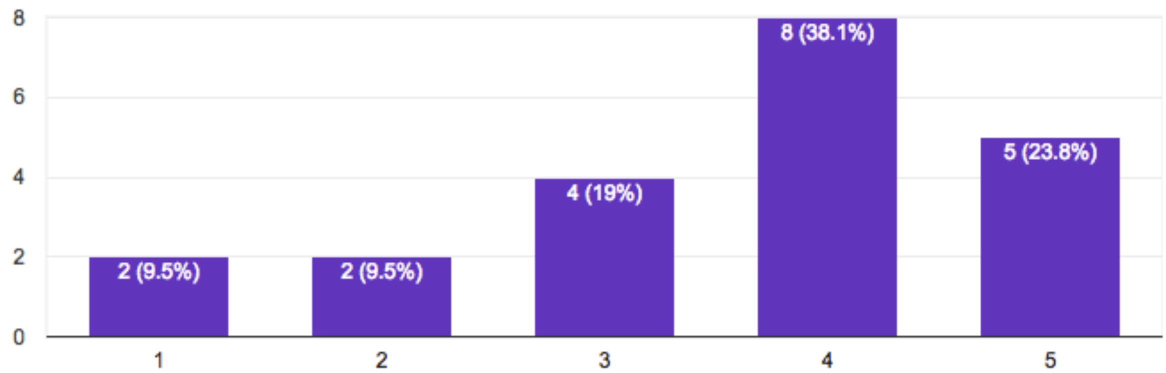
Useless - useful

21 responses



Lacking - extensive

21 responses



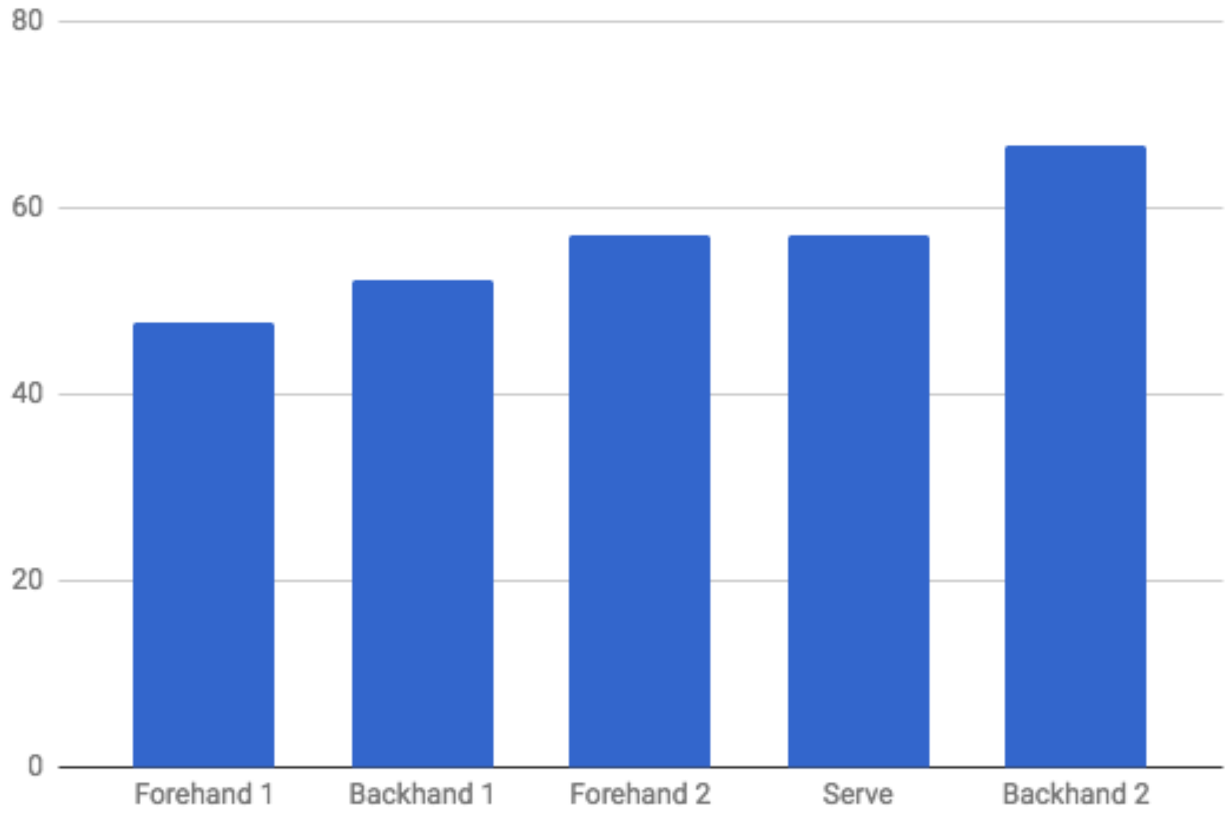
The users
Comments regarding this topic:

- At the moment this would for me be an application that you would use with your trainer, and maybe alone after a while of working with it.
- I still think that this should be used with a trainer and should not replace the conventional trainer.
- I think this would be most useful for professional tennis coaching with a coach watching the screen.
- It was hard to recognise the kind of swings and to give scores because of my lack of tennis experience. However, I think that the software will be useful for people with tennis experience
- I see that this can be helpful for professionals, but I don't think that regular tennis players would use it. Maybe if tennis clubs have this application and tennissers can borrow it, it would be used.

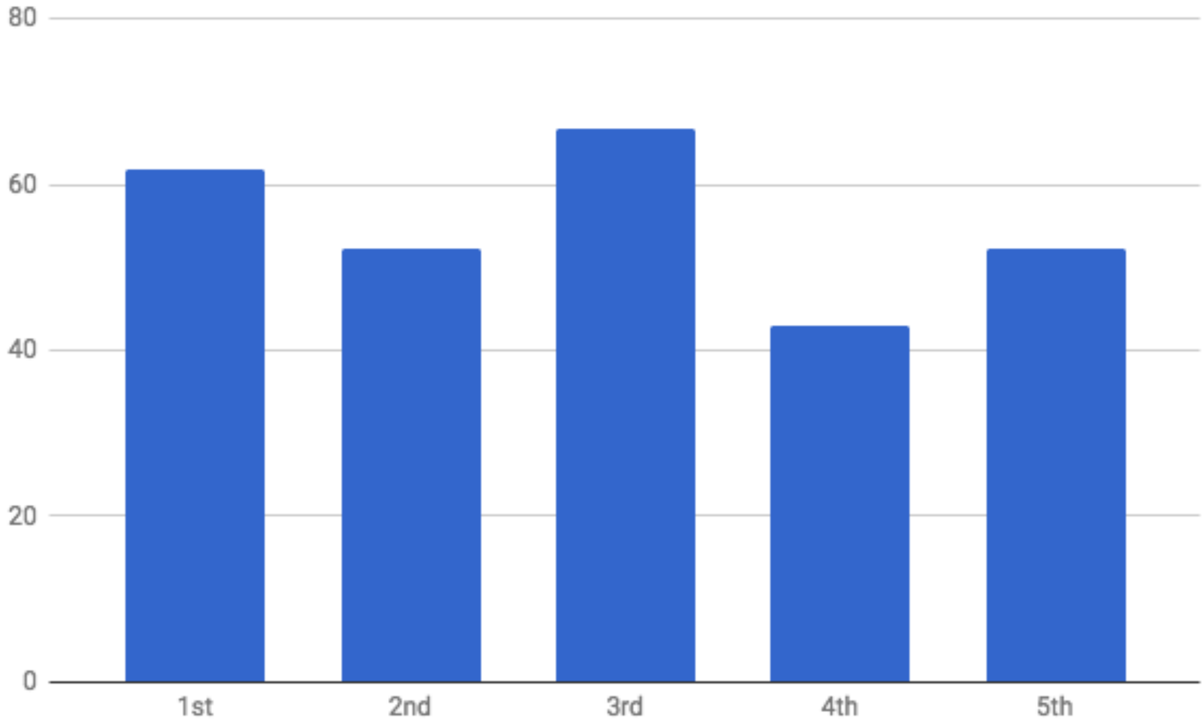
Sound
Comments regarding this topic:

- Maybe a sound feedback could be used to indicate the swing movement.
- Sound feedback in the menu (favourite and play/pauze buttons) could make it better understandable. I did not notice that my third swing was added to the favourites, this is why I added it twice to the list. Maybe a sound could be played when adding a swing to the favourites.

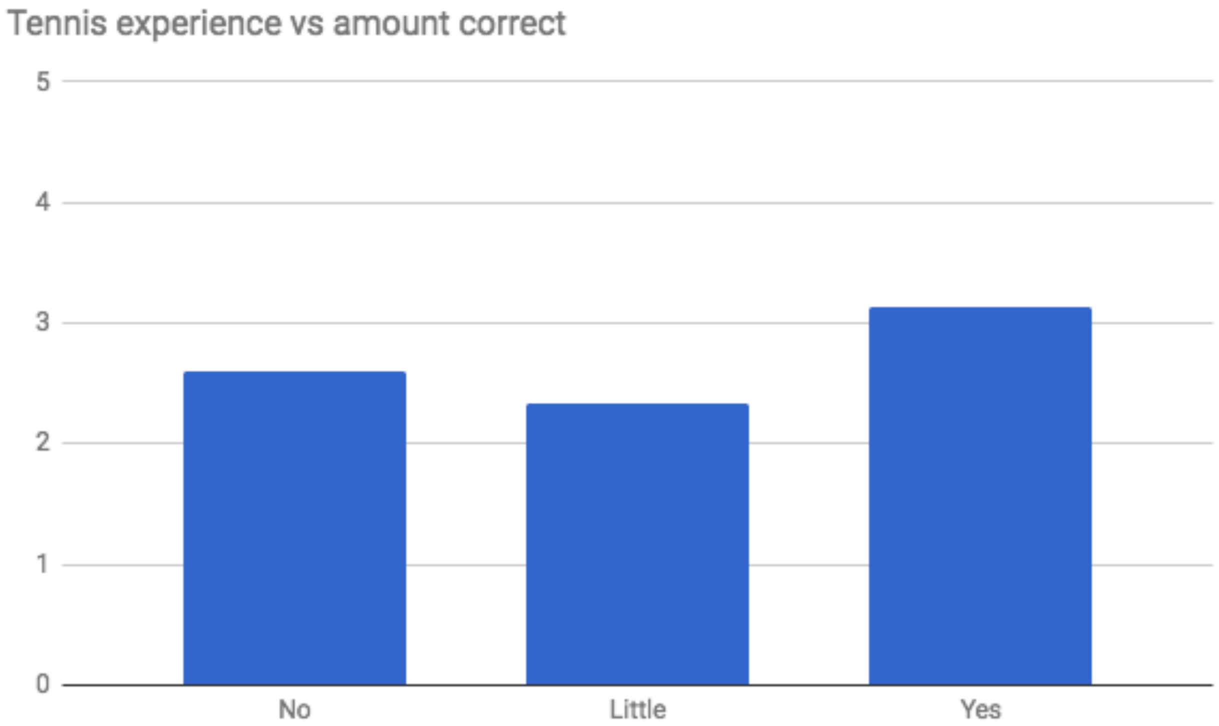
Identifying swings



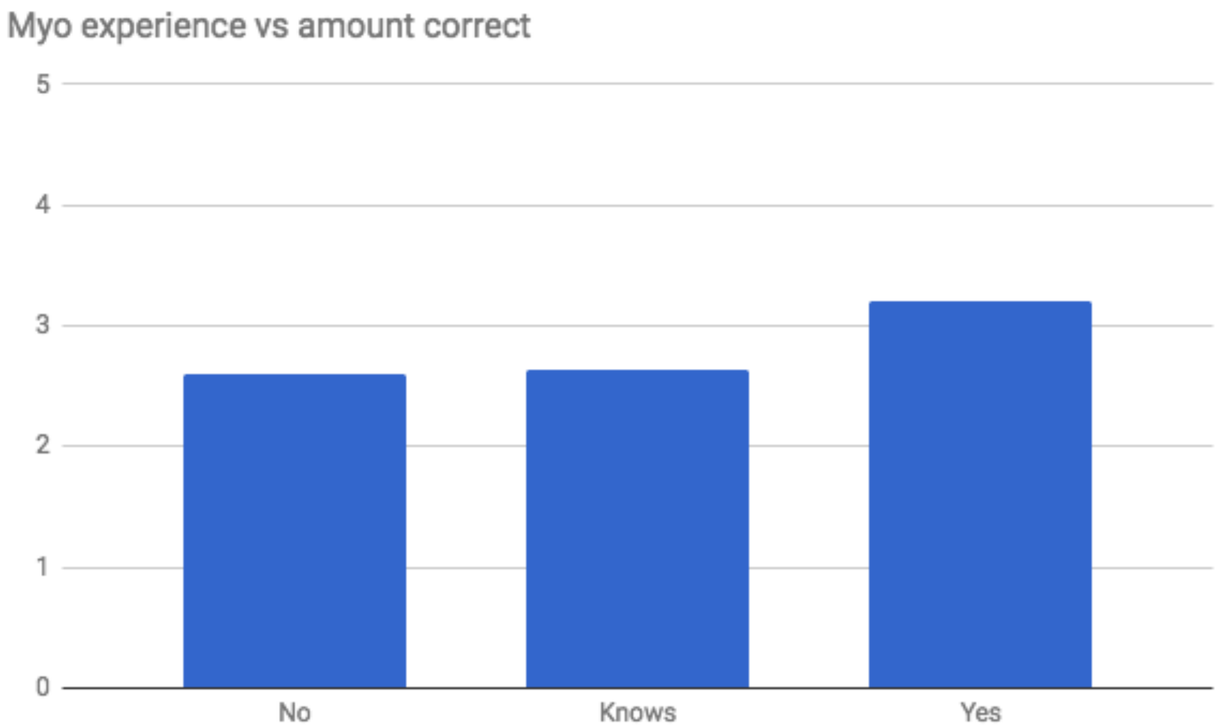
Percentage of the correct answers of the different types of swings people were asked to identify.



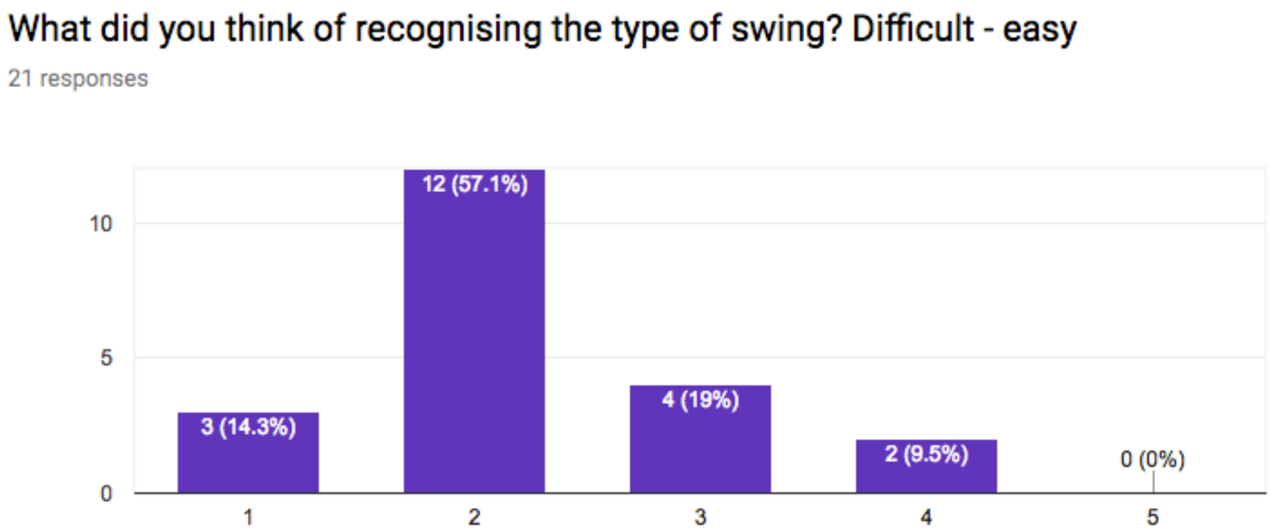
These are the percentages a swing was identified correctly based upon which question it was.

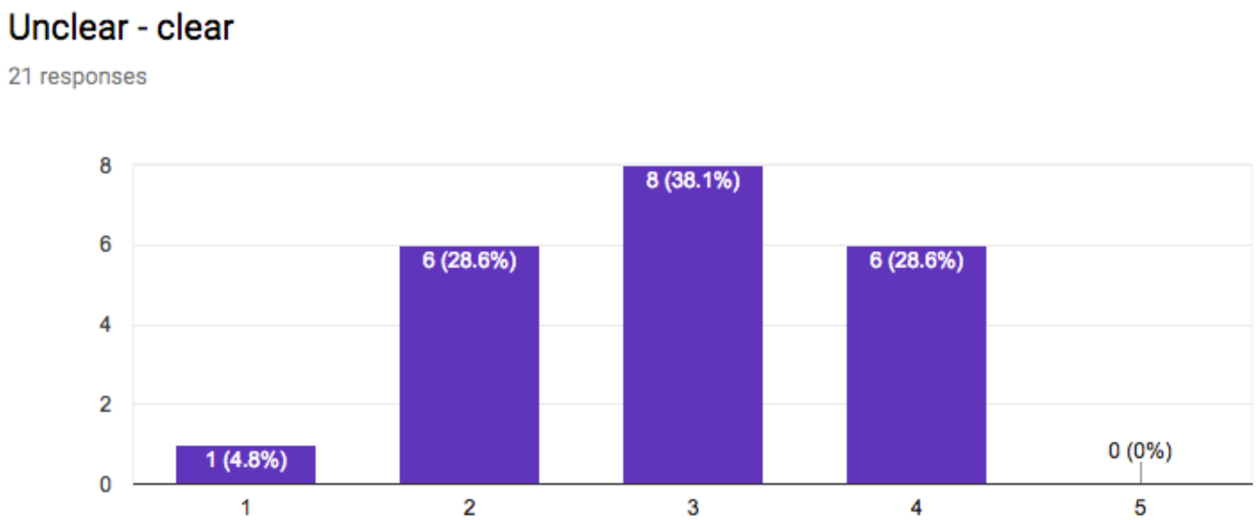


The relationship between tennis experience and average amount correct was also looked into



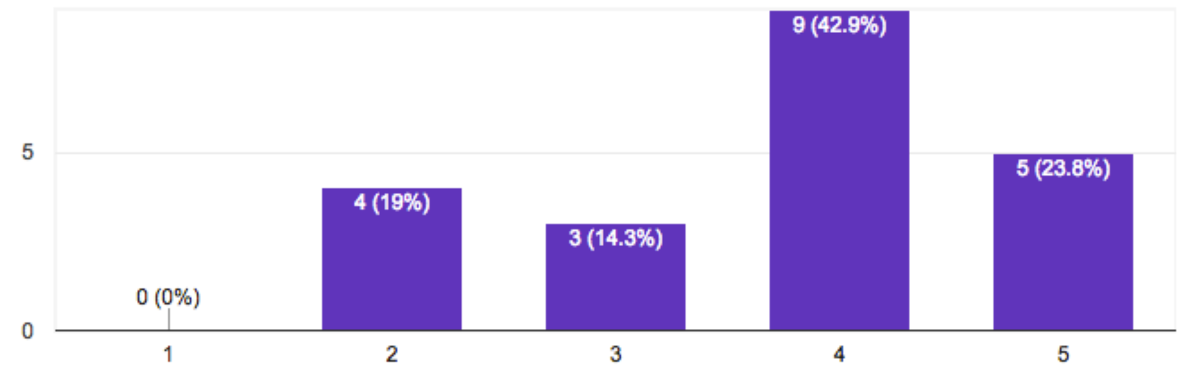
Also the relationship between Myo experience and average amount correct.





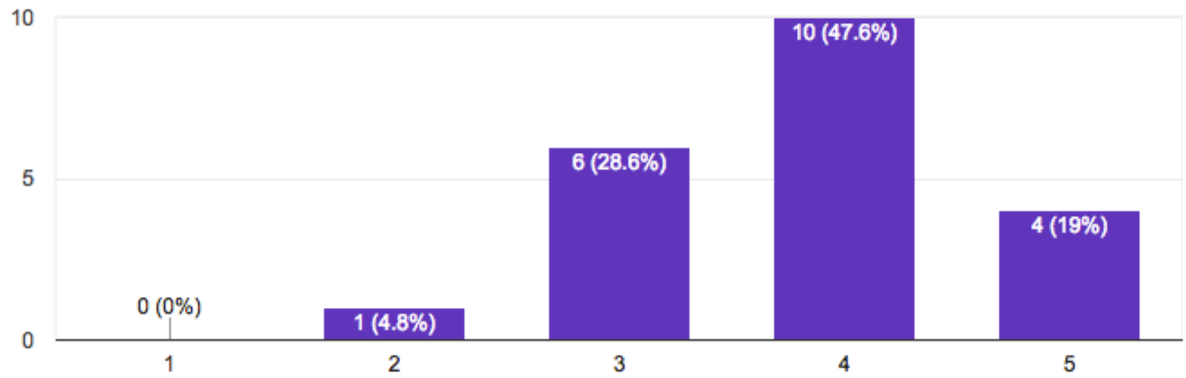
What did you think of recognising the amount of swings? Difficult - easy

21 responses



Unclear - clear

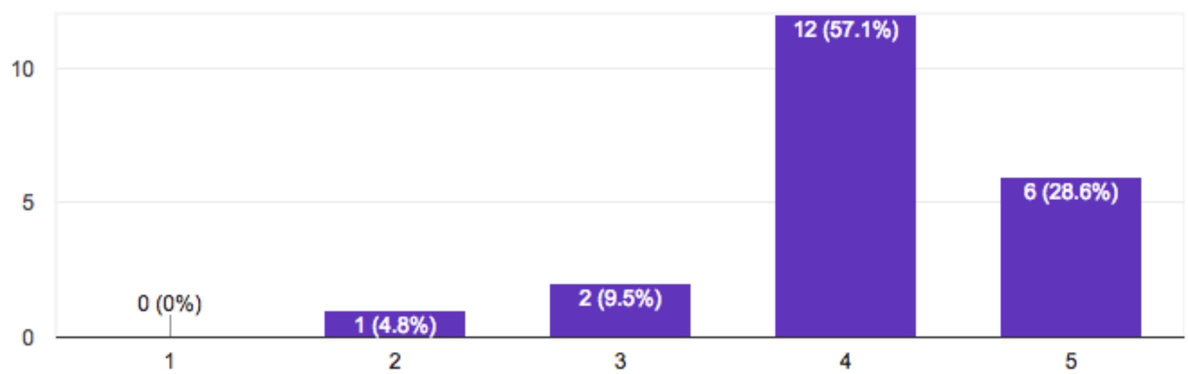
21 responses



Tennis

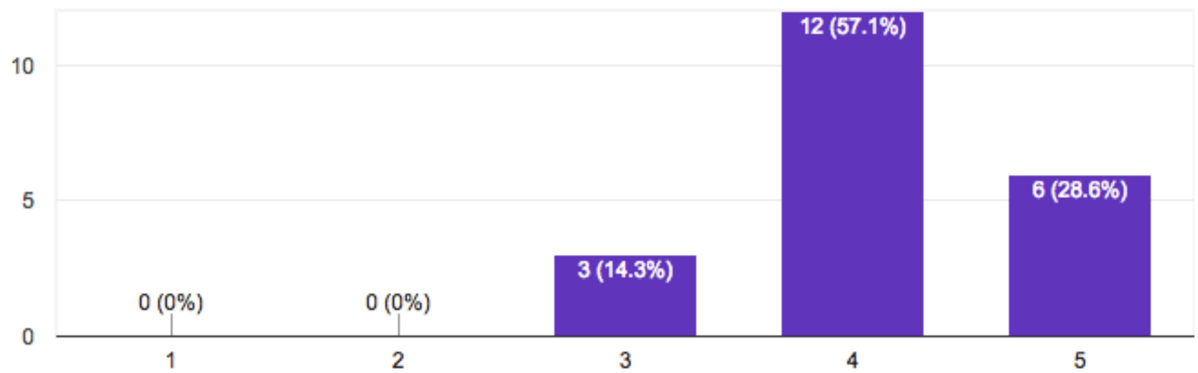
What do you think of this program in regards to tennis? Useless - helpful

21 responses



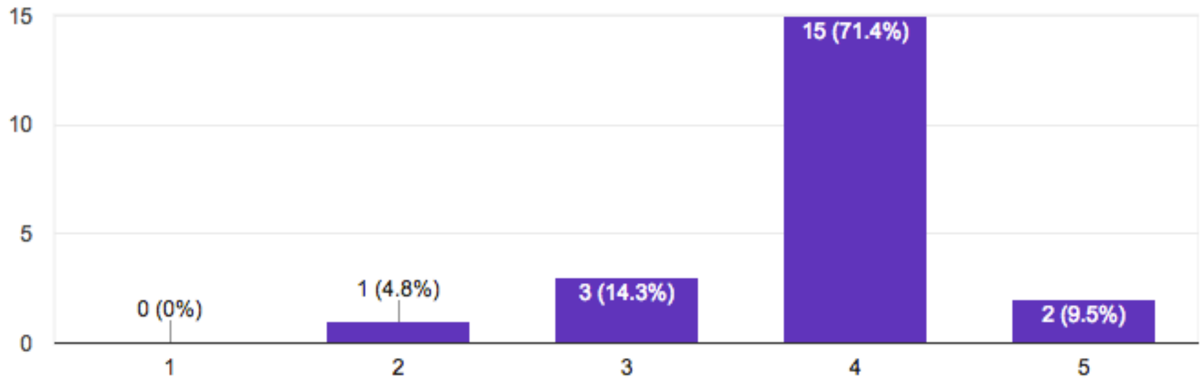
Unsuccesful - addition

21 responses



Ineffective - effective

21 responses



The grade

What grade would you give this application?

21 responses