University of Glasgow | School of Computing Science

Physiotherapy Feedback Using the Microsoft Kinect Motion Sensor

Velizar Shulev

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

Level 4 Project — March 27, 2015

Abstract

Rehabilitation after an injury involves regular sessions with a physiotherapist, who is there to assess the patients' condition and to advise a course of action. This course of action involves a personalised exercise programme which the patient needs to follow at home. Sometimes this is problematic, because patients may not have motivation or otherwise may be performing their exercises poorly. This report describes how a system was designed, which, using the Microsoft Kinect sensor, monitors and assesses patients, while they are exercising at home. The goal is to provide real-time feedback to patients, while simultaneously reporting to the physiotherapist. The feasibility of the system is evaluated by testing some of its components.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: _____ Signature: _____

Contents

Chapter 1

Introduction

Part of the rehabilitation process after an injury involves regular sessions with a physiotherapist. During these sessions patients get an assessment of their condition and are given exercises to perform at home. This is where the majority of the rehabilitation takes place. This can be efficient for physiotherapists, who have time to tend to more patients. For patients, however, home rehabilitation may be problematic. It's not uncommon for patients to suddenly lose motivation to exercise, or alternatively, to perform exercises with poor form without realising it. The recovery process can be significantly hindered by these circumstances.

This report examines the design of a system, which, using the Microsoft Kinect sensor, will monitor and assess patients, while they are performing exercises at home. At the same time physiotherapists will be notified about their patients' activities over the network.

1.1 Main Components

This report describes the design of an application for home rehabilitation, which consists of 3 components.

The first component is an application which will be run on a desktop machine at the patients' homes. The application will be connected to a Kinect sensor in order to recognize and track exercises done by the patient. It will also provide feedback. Finally, the application should send data back to the physiotherapist, in order for them to track the patient's performance and react promptly if there are any issues.

The second component is an application for physiotherapists, which will be heavily reliant on effective data visualizations. The application also needs to play back recorded data of patients performing their exercises at home.

The final element of the system is a repository of exercises and a data store for patient recordings. The patient application needs to retrieve details of exercises assigned by the physiotherapist and needs to submit recordings to the data store. The physiotherapist application needs to retrieve these recordings in order to visualise the patient performance.

1.2 Report Overview

The remainder of this report is divided into the following chapters:

- Chapter 2 provides some background to the problem, briefly explains how physiotherapy works and defines a few important terms
- Chapter 3 provides an overview of related works and analyses how the information can be used in this project.
- Chapter 4 is dedicated to the design of the proposed system
- Chapter 5 describes interesting aspects of the implementation work which was conducted as part of this project
- Chapter 6 provides information about how the project was evaluated and describes some important results
- Chapter 7 concludes the report, details the insights gained, the challenges and proposes future work.

Chapter 2

Background

This chapter explains the concepts and terminology needed in order to understand the rest of the report.

2.1 Physiotherapy

The physiotherapy process starts with an initial examination by a physiotherapist, who assesses the extent of the physical disability. Consecutive sessions will then occur, which at first may happen several times a week, but as some recovery is observed would become less frequent. During these sessions the physiotherapist will work with the patient by setting goals, which may start simple (for example picking up and object) and become more complicated as the treatment progresses. Even though treatment can take place at a clinic, home-based physiotherapy is quite common. The physiotherapist will assess the patient and prescribe an personalised exercise programme during each session. The patient will then need to follow the programme at home [17].

2.2 Exercise Evaluation

In order to recognise exercises, evaluate performance and give feedback to a patient, good insight is required regarding how physiotherapists asses exercises. For this purpose 5 physiotherapists were consulted initially. In addition, one physiotherapist was regularly consulted throughout the project. The following model of representing exercises was developed after discussions with physiotherapists.

2.2.1 Exercise Components

In order to be effective each exercise needs to be performed a given number of times in a row. A single attempt is called a repetition. Repetitions are grouped into sets. Between sets, patients have to take a rest, the length of which is determined by the physiotherapist.

Four major aspects of an exercise were identified, where patients are likely to make mistakes. These in addition to sets and repetitions are referred to as exercise components (or just components) throughout this report. This is not a term used in physiotherapy, as these components are conceptually different from each other, but treating all of them in a similar way allows for easier modelling of the system. Furthermore these components have been divided into two categories - quantitative - i.e. those that can be assigned a count and qualitative - can be assigned a binary value - either ”good” or ”bad”. These six components are continuously assessed in real-time,

while a patient is performing an exercise, so their values will change over time. The components of an exercise are as follows:

- Progress (qualitative) indicates how much progress the user is able to make in the repetition. This component is given a binary value based on some empirically determined threshold value.
- Speed (qualitative) indicates whether the user is doing repetitions at the correct speed.
- Control (qualitative) indicates whether there is too much variation in the speed of the user or whether speed is uniform throughout the repetition.
- Posture (qualitative) indicates whether the user's joints are correctly aligned while performing the repetition.
- Repetitions (quantitative) indicates the number of times the exercise has been performed within the current set.
- Sets (quantitative) indicates the number of sets which have been performed.

Most components are assigned a binary value instead of some continuous score, because finding algorithms to quantify them is challenging. Furthermore it's uncertain what benefits continuous values will have to users. Telling users their speed is "bad" makes much more sense than telling them their speed is 6.558934.

2.2.2 Mistakes

A mistake is defined to occur in a given component, when the value of a component is set to "bad".

2.2.3 The Arm Abduction Exercise

A simple exercise needed to be chosen in order to test the ideas in this report. Few exercises were considered before finally choosing the arm abduction. An armed abduction is performed by lifting one's arm to the side until it is parallel with the floor. The arm is then lowered back to its initial position.

2.3 Current State of Technology in Physiotherapy

Currently the most common way for a physiotherapist to prescribe a personalised exercise programme is to give the patient a printed handout of exercises. Handouts are noramlly produced by an application called PhysioTools [15]. After the patient leaves the session, there is no way for the physiotherapist to know whether their patient is adhering to the programme. Furthermore, the patient cannot receive any feedback on the quality of their performance.

2.4 Potential Users

Potential users of the application, described in this report, include everyone undergoing some form of home-based physiotherapy, as well as most physiotherapists.

However many patients are elderly people with reduced mobility as a consequence of their condition. Therefore care should be taken to ensure elderly people will be comfortable using the application.

2.5 The Microsoft Kinect

The Microsoft Kinect [11] is a line of motion sensors for Xbox 360, Xbox One and Microsoft Windows. Kinect is equipped with a camera, a depth sensor, a microphone and various other sensors. It can track up to 6 people simultaneously, providing up to date joint data, which includes both joint position (in 3D space) and joint orientation (a quaternion).

Kinect comes with a Software Development Kit (SDK) for Windows, which provides easy access to raw sensor data, as well as processed body data (joints). The SDK also provides a default interaction style, which involves hand gestures. Instead of the regular mouse pointer, a hand cursor is displayed on the application screen, which is controlled by the user's hand movements.

Kinect has had two major releases to date - the V1 and V2.

2.5.1 Kinect V1

The V1 was initially released in 2010. It can track 20 joints.

2.5.2 Kinect V2

The latest version of Kinect - V2 improves on its predecessor. It is capable of tracking 26 joints (Figure 2.1) and offers a higher resolution. The V2 SDK also comes with a few improvements.

2.5.3 VGB - Gesture Recognition

One of the new features of the V2 SDK is the Visual Gesture Builder (VGB). It is a set of tools and libraries, which enable developers to easily implement gesture recognition in their applications. VGB uses various machine learning algorithms in order to detect when a gesture is being performed.

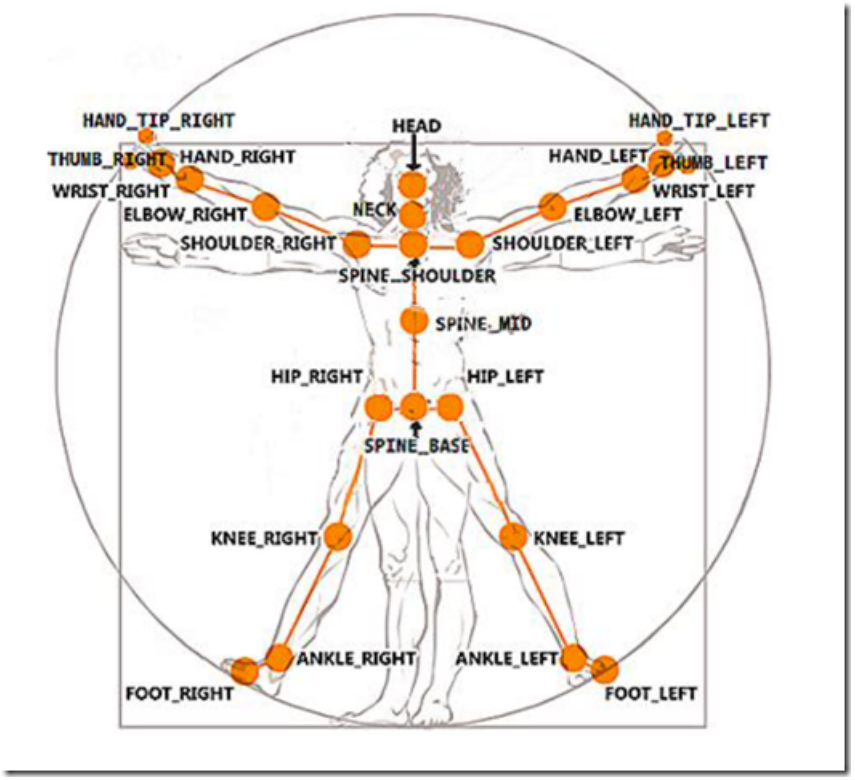

Figure 2.1: Kinect V2 joints

Chapter 3

Related Work

Despite PhysioTools still being the most common technology used in physiotherapy, using motion sensing devices for rehabilitation has been a hot topic for a while and there is a wide body of existing research.

Due to the goal of this project being to evaluate the feasibility of an end-to-end system for monitoring and assessment of patients, different areas of research have been examined.

3.1 Exercise Recognition

One major area of research is exercise recognition. In many ways exercise recognition is similar to gesture recognition. Gesture recognition algorithms for the Microsoft Kinect analyse joint data in order to determine whether a gesture is being executed or not. Unlike with gestures, however, with exercises it is not enough to just detect whether a user is performing them or not. Feedback also needs to be given to users regarding their performance. In other words, for this specific project, exercise recognition includes not only detecting the exercise but also evaluating the 6 exercise components.

Research in the area of exercise recognition falls under two broad categories - rule-based recognition and machine learning recognition, depending on the particular set of algorithms used. The following sections highlight the differences between the two approaches, taking into account previous papers on the topic.

3.1.1 Rule-based Recognition

Rule-based recognition (term has been used in [24]) uses explicit rules to define a gesture or an exercise. An example of a rule is - the angle between the bones starting at the elbow needs to be between 30 and 40 degrees. The same rule is demonstrated in code in Listing 3.1. A gesture usually consists of multiple rules, which are ordered in some sequence. The user must follow the rules in the specified sequence in order for the system to recognise the gesture.

```
1 if(elbowAngle >= 30 && elbowAngle <= 40)
2 {
3     doingIt = true;
4 }
```

Listing 3.1: Defining a rule in code

The advantages and disadvantages of this approach have been outlined in [20] and [31]. Rule-based recognition is good for the following reasons:

- No training data is required for the algorithm to work. Collecting and classifying data can be a time consuming task.
- Adding a new gesture does not require recording additional training data and testing data.
- Interpreting a gesture comes naturally to every person. Defining the rules for a given gesture comes easily to most people.
- There is no need for careful statistical and mathematical modeling. There is no risk of overfitting.

However, rule-based recognition has serious limitations:

- Rules are defined as part of the code, which need to be implemented by someone with programming knowledge.
- For each rule thresholds need to be manually found and tuned.
- Does not generalise well to a wider variety of people and environments. Small changes to external variables may necessitate adding additional rules to the gesture.
- Latency decreases as more rules are added and code grows larger. Sometimes previous frames may need to be examined as well, which can hinder performance even more.

The issue of this approach not being accesible to non-engineers is solved by the Gesture Description Language (GDL) introduced in [31]. GDL attempts to simplify rule-based recognition, by introducing a simple language designed specifically for defining gestures. [31] claims that the rule-based approach is sufficient for most cases, with recognition being successful 80.5% - 98.5% of the time. Other papers also propose a custom language for defining rules [37]. The problem with GDL and the rule-based approach in general is that mistakes cannot be detected, with regards to the 6 exercise components model proposed in this report.

The rule-based approach is very common and can be encountered in a few papers such as [35], [29], [37] , [34]. In [35] the authors admit a limitation of their system - some of the rules may have been too strict and required some fine tuning.

3.1.2 Machine Learning

Another approach to gesture and exercise recognition involves using classification algorithms. There are many papers which have tested the feasibility of machine learning algorithms for recognition, for example [24], [30]. The conclusions are that it is generally effective.

The advantages and disadvantages of using Machine Learning for gesture recognition have been described in [20]. Some of the benefits include:

- Regardless of the size of the training data, the output values from the training usually have the same size and therefore performance when predicting new values is fairly constant.
- Training requires labelling some raw data and the supplying it to the algorithm. No programming knowledge is required after the algorithm has been implemented, therefore anyone can train it, physiotherapists included.

- By providing varied training data the algorithm can adjust to variations in users (such as body types, clothes) and the environment (such as sensor positioning and tilt angle).

The drawbacks of using this methods are as follows:

- It is time consuming to record and label training data.
- Adding new gestures is therefore slow and potentially expensive if paid participants need to be recruited.

An additional risk with machine learning is overfitting - that is making very good predictions for the training data, but making many errors when testing the algorithm with new data.

Unlike the rule-based approach, machine learning can detect mistakes with regards to the 6 components defined, by using the Visual Gesture Builder tool.

3.2 Providing Feedback to Users

Another important aspect of any home-based rehabilitation system is how to effectively provide feedback to patients. Patients need to be made aware of the mistakes they are making, so that they can correct them.

3.2.1 Visualising the Exercise

There are two ways of displaying the current exercise to a user. One way includes displaying overlay elements on top of the Kinect camera feed. This is a common approach used in several papers, amongst which [29]. This may be regarded as an interface metaphor, as sometimes physiotherapists recommend doing exercises in front of a mirror in order to pay attention to proper form.

An alternative method is to use avateering (see Section 5.5). Avatars have been used in [28] and [37]. They are also popular in exergames (Section 3.2.3) - for example [33], as these often take place in a fictional setting. Some of the reasons according to [28] and [37] include maintaining patient anonymity, placing the patient and their physiotherapist in the same virtual space and enabling users to observe their activity frame-by-frame from any viewing angle.

No evaluation results for any of the two methods were found, therefore testing either one of them would be valuable for future research.

3.2.2 UI Elements

A variety of UI features can be used to provide real-time feedback to users about their performance. Elements can vary from a progress bar which fills up gradually as the user performs the correct movement [29] to background music [35], [36] which can be used to better engage users. Highlighting parts of the model is also an option [37].

Informing users about their long-term performance is also important and can be done using diagrams or charts [23]. The performance shown is in terms of the quality of exercise performance as well as the average number of repetitions per week.

3.2.3 Exergames

Exergames are a category of video games which are also a form of exercise. Exergames are made possible thanks to technologies which track body movement such as the Wii Fit [21] and the Microsoft Kinect. In rehabilitation exergames patients are not asked to do any specific exercise, but are rather encouraged through gameplay to perform certain movements, which have been approved by physiotherapists. A simple exergame revolving around collecting virtual coins is described in [33]. It was tested with end users who gave positive feedback about the game. An advantage of exergames is that they can use techniques found in other types of games in order to keep users engaged. Collecting virtual currency is a common theme in freemium games [26], which are notorious for maintaining high user interest.

Another exergame is presented in [25]. This one, however, has not been evaluated with users.

The biggest disadvantage with exergames is that users do not perform actual exercises. This contradicts the established practice of home-based physiotherapy where patients are given a specific exercise programme, which they need to follow. Therefore exergames can not be considered a replacement of the traditional methodology. Concepts from exergames and games in general, however, can be worked into an application which enables home-based rehabilitation. For example, users can be given a score after each exercise or they can be awarded points for good performance, which can be somehow spent later.

3.3 Input

No papers, relating to physiotherapy with the Kinect, describe specific input methods. [23] describes a system which uses a remote control, and concludes that it is a good method. Kinect offers a default input method based on gestures, which has not been evaluated in any of the existing projects (to the best of my knowledge). Speech is also a possible alternative.

Due to the lack of any reported results, evaluating any of the mentioned input methods would be useful to look into. Based on intuition a good starting point may be the Kinect default input method, because it has been used in most Kinect games, so a hypothesis can be formed that it will be efficient.

3.4 Hardware

In many of the papers mentioned in the sections of this chapter Kinect was used to evaluate exercises (for example [33], [35], [37]). Other hardware has been tried and tested for rehabilitation too [23], [30]. Unlike these sensors, however, Kinect doesn't need to be attached to a user's body and requires very little initial setup. The question, whether Kinect is feasible for home rehabilitation is answered by the overwhelming amount of available publications.

3.5 Companies and Products

There are already some companies offering products similar to the projects described in this chapter. Companies offering rehabilitation software include Reflexion Health [16], VirtualRehab [19], JinTronix [5]. Although they may have functionality similar to the project described in this report, the algorithms and methods they use are kept private, so no insight could be gained from them.

Chapter 4

Design

The design of this project was guided by existing literature on rehabilitation software, as well as papers and articles related to user interface design.

4.1 Requirements

This section outlines the main requirements for this project, gathered through frequent meetings with a physiotherapist, as well as two focus groups with 5 physiotherapists. Standard software engineering practices dictate requirements gathering should be a rigorous process, throughout which requirements are consistently refined and broken down into more manageable and better defined chunks. For the purposes of this project, it was decided to only define the higher level requirements. Smaller design decisions (such as choice of icons or particular user interface elements), which were made as a result of feedback from physiotherapists, are not listed in this section for the purpose of keeping it concise.

4.1.1 Functional Requirements

The following were identified as the main requirements for the system. They have been split into two groups, based on whether they are related to the patient application or the physiotherapist application (including the data stores). See section 1.1 for a brief summary of the main components of the application.

Patient application

- The application should be able to recognise a simple test exercise. The chosen exercise, as well as the reasons why it was chosen are described in Section 2.2.3.
- The application should be able to recognise the components outlined in Section 2.2, rate them as ”good” or ”bad” if they are qualitative, or track their count if they are quantitative.
- Patients should be notified in real-time of any mistakes they are making.
- The movements of the patient should be mirrored by a 3D avatar. This enables users to observe their motion from potentially different viewing angles, which in turn can lead to a better understanding of why they are making mistakes and a better learning experience.

- Users should be able to interact with the application from a distance. This means the traditional input method - using mouse and keyboard - is not suitable for the purposes of this application and alternative options need to be explored.
- The application should be able to track how well the patient is following their assigned exercise programme.
- The user should be kept informed about how well they are performing their exercise programme.
- The application should send patient data to the physiotherapist application.

Physiotherapist application

- The application should visualize information collected from patient applications, so that a physiotherapist can monitor their patients' performance.
- The application should store all patient data it receives, so that past records can be examined.
- Physiotherapists should be able to replay an exercise done by a patient exactly as it happened. This is to be done with the help of a 3D avatar. Physiotherapists should be able to view the avatar from different angles.

One important aspect has been omitted from the requirements - real-time communication between a patient and their physiotherapist. [36] identifies long distance between patients and their physiotherapists as a potential barrier. Video-conferencing functionality would enable patients to attend virtual sessions. This requirement would be out of the scope of the current project. The aim of this project is not to replace or reduce physiotherapy visits. Therefore real-time session functionality may be useful, but is an not essential requirement.

4.1.2 Non-functional Requirements

The following list contains the main non-functional requirements for the distributed system.

- The patient application should run on an average mid-range desktop computer. This reduces costs if machines are to be distributed along with the application across patients' homes. It also means that more patients will have access to the application if they have to install it on their personal machines.
- The algorithms for assessing exercise performance in real-time need to be efficient, otherwise the system may suffer from latency issues.
- Sending data from the the patient application to the main server has to happen in the background. Users must not have to wait for the application to finish sending data, before they are able to interact with it.
- Data should not be kept on the patients' machines for too long, otherwise the local cache may get too big and new data will not be stored.
- The system should be scalable. Patient applications are likely to send large amounts of data over time. The databases used in the systems should be able to handle large quantities of data.

4.2 User Personas

Requirements are not enough to completely understand how the system should be designed. Knowledge of the users is essential in order to make appropriate design decisions. For this purpose two user personas were invented, following discussions with physiotherapists and examining the fictional users developed in [36]. The main principles of interaction design dictate that user personas be created after interviewing a significant number of users [27]. The physiotherapist persona was created after talking to a small number of physiotherapists. The patient persona is not based on any interviews with real patients but rather on descriptions gathered from the physiotherapists. The approach to defining user personas in this report may therefore have limitations, mainly, there is a risk that users needs and goals were not fully understood.

The two user personas - one representing a patient and the other a physiotherapist - are described in the following sections.

Judith

Judith, 81, has recently suffered a stroke and is experiencing weakness in right side of her body as a result [17]. She is just starting physiotherapy and is about to visit her physiotherapist for her first consultation. In order to recover she needs several sessions a week [17], but due to her reduced mobility, visiting the clinic is difficult. As a result of the stroke, she is also suffering from extreme tiredness [17] and as a result may find it difficult to find motivation for exercising. Judith has heard that she may be assigned a care worker [17] and is hoping to get one, because she feels like she won't be able to cope on her own.

Simon

Simon, 43, is a physiotherapist with 15 years of experience working with stroke patients. Simon has been disheartened lately by the growing lack of motivation he is seeing in his patients. It is common practice in his clinic to use PhysioTools, however he suspects that printed exercise handouts may be confusing to some patients, who are not able to understand some exercises and therefore lose motivation. Recently Simon has created a YouTube playlist of exercise videos he has been able to find. He has noticed that many of the elderly patients know how to browse the web, so he sometimes gives them links to the videos. A few patients have reported they are quite happy with the videos. Simon has two children, a boy aged 10 and a girl aged 14. His kids play regularly on their Xbox One console. Last Christmas, they received a Microsoft Kinect as a present. Having personally tested the capabilities of the new motion sensor, playing video games with his children, Simon has been thinking whether the Kinect would be any good for physiotherapy.

4.3 System Architecture

Based on the formulated requirements the overall system architecture, as well as the architecture of each component were designed. A diagram representing how the final system should look like is presented in Figure 4.1. An entire system was not developed in accordance with this diagram. The only implementation work was done on the patient-side of the system. The diagram, however, stands as a reference for what a complete system should look like.

A similar architecture has been proposed in [24]. The difference is that this report explicitly defines where the servers and data should be located, which is an important detail when it comes to developing a real world system. This report proposes that an Infrastructure As A Service provider should be used for hosting application servers

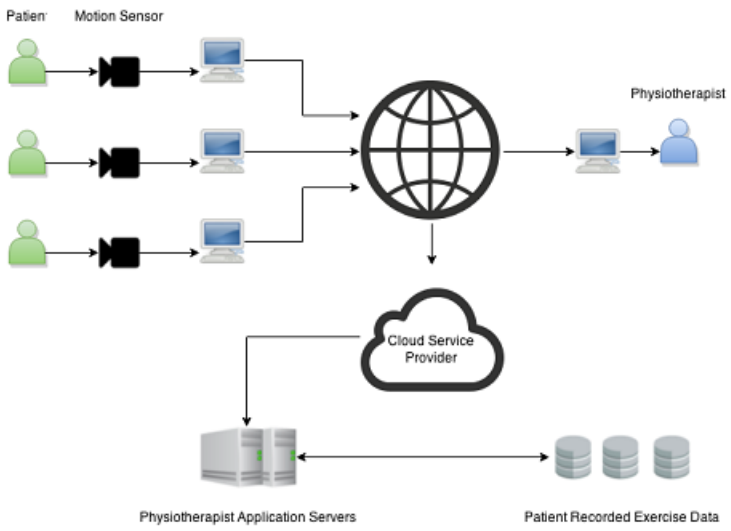

Figure 4.1: Diagram representing the entire system.

and databases. Large cloud service providers, such as Google [4], Microsoft [10] and Amazon [1], provide excellent scalability. If the proposed system is rolled out, it is likely to go through a pilot stage with a small group of users. Dedicated servers in the clinic will be too costly. Cloud services, on the other hand, provide a model in which cost scales with usage, therefore initial expenses will be lower, making the proposed system more attractive.

4.3.1 Patient Application

The architecture for the patient application has been illustrated with the help of a UML class diagram (Figure 4.2). The final implementation is still very much a prototype so it does not follow the structure presented in the diagram.

Class Structure

The application classes are split into 3 packages.

The package SensorReaders contains two classes GestureReader and BodyDataReader. They serve as adpaters to the Kinect SDK. BodyDataReader reads joint data from the sensor and notifies the application when data is updated. Similarly GestureReader notifies the application on update, but it reads in gesture data with the help of the Visual Gesture Builder (see Section 2.5.3). Both classes are designed as singletons - only one instance of each is available during application runtime. This reduces coupling in the application, because references to instances of the two classes don't need to be passed around as parameters.

The Exercise package contains implementation details specific to exercises. It contains the classes for each exercise, which in turn contain methods for evaluating each component (it is assumed that component evaluation may vary between exercises).

Finally, the UserInterface package contains all classes related to the user interface. Most classes along with their corresponding .xaml pages represent Windows Presentation Foundation (see Section tools and languages) pages. Classes related to rendering the 3d scene and the avatar are contained within the Graphics subpackage.

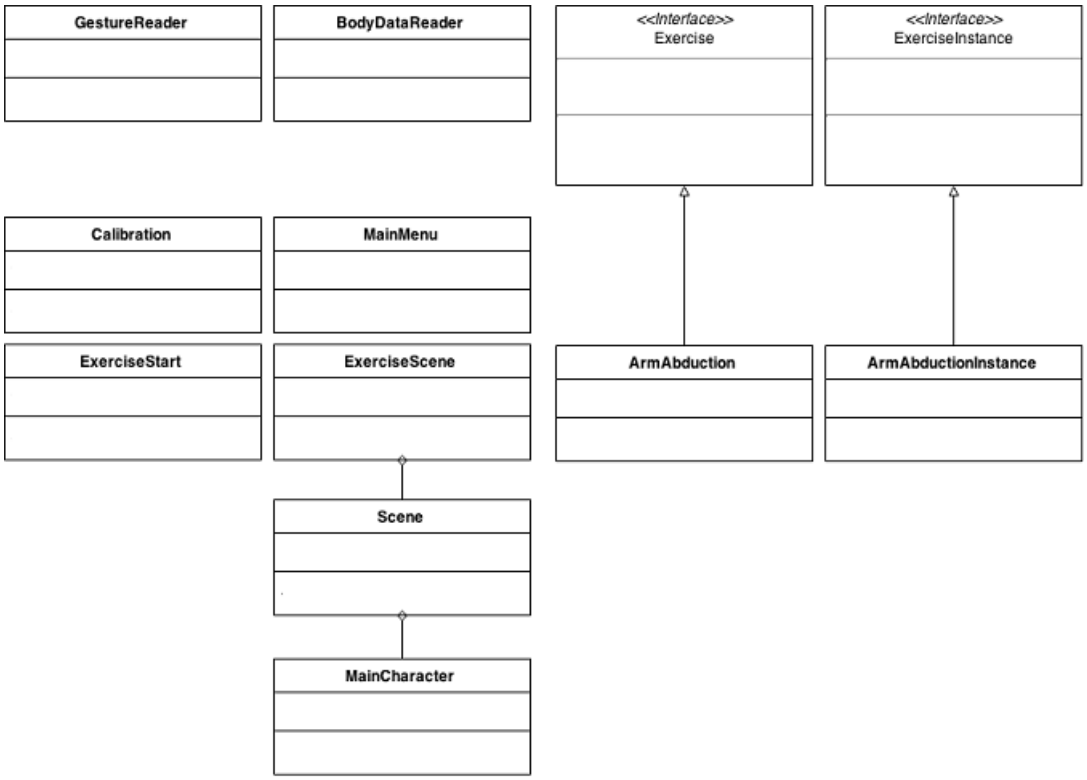

Figure 4.2: UML of the patient application.

Design Patterns

Design patterns have been used throughout the application in order to reduce coupling. As already mentioned one of them is the singleton pattern. Another one is the Observer-observable pattern, used in multiple classes. It helps create a layered structure. At the bottom are the classes which read data from the sensor. They notify the classes above them, which in turn notify the classes above them. This creates a clean separation between the layers of the application. Each layer doesn't need to know anything about the layers above it.

Extensibility

The application offers some degree of extensibility with regards to implementing new exercises. Each exercise comprises of 3 parts - an entry in the gesture database (see some section in implementation) and two new classes. Updating the database can be done by anyone, not just software engineers. The classes which need to be created must extend Exercise and ExerciseInstance. Both classes are described in more detail in Section 5.2.

Improvement Suggestions

The final implementation presented in this report is still far from a finished product. Therefore there is a lot of room for improvement. One issue is that the current architecture does not support adding new components. It is assumed that each exercise will use the same 6 components. Another design problem is that adding exercises involves writing code. The task of adding exercises cannot be done by physiotherapists or other members of staff working at the clinic. The exercise model should not include any classes, only data files. Finally, there are no classes for handling communication with the exercise servers.

4.3.2 Physiotherapist Application

Based on feedback from physiotherapists, the consensus was that the physiotherapist application will be used from a desktop computer located in the clinic. A desktop application therefore seems like a viable option.

An alternative option is to develop it as a web application. This approach has a few advantages. First and foremost a web application is platform agnostic, meaning it can run on any operating system and a wide range of browsers, making compatibility less of an issue. Secondly, the application does not need to store data on a local machine - it only needs to pull data from a server and visualise it. Therefore the functionality provided by the operating system is not needed. Finally, there are many mature web application frameworks [6], which are able to produce single-page web applications, which behave similarly to traditional desktop applications.

4.4 Design Process

The design process followed the iterative design methodology, where each iteration consists of 3 stages - gathering and analysing requirements by talking to physiotherapists, coming up with prototypes and finally evaluating these prototypes. This process was repeated 3 times. The effects of the iterative design process are a better understanding of the core requirements and the refinement of some of the major features of the application.

Evaluation is briefly discussed in this section. More detailed results are presented in Chapter 6.

4.4.1 First Iteration

Requirements

The main goal of the first iteration cycle was to develop a better understanding of how exercise feedback should be provided to patients. Initial discussions with one physiotherapist revolved around whether displaying the video feed captured from the Kinect camera with some information overlaid on top is preferable to avateering (see Section 5.5). These options were communicated to the physiotherapist through images and videos found online. The decision was made to use avateering, as it will enable switching the viewing angle in the rendered scene based on which exercise is being performed. Seeing themselves from optimal angles will allow patients to get a better understanding of their mistakes. An additional point was made, that some patients may dislike staring at their own image while doing an exercise, and therefore will prefer to have an avatar. Interviews with patients need to be conducted in order to confirm or deny this.

Prototype

No paper prototypes were designed at this stage, because no UI had been discussed yet. A paper prototype would not have been suitable for demonstrating avateering due to the inability of users to directly interact with the 3d model. A simple exercise was chosen - a bicep curl - to demonstrate exercise recognition. A nave rule-based approach was used in order to detect the start- and mid-points of the exercise. Only sets and repetitions were tracked in this demo. Every time a repetition is completed a short sound is played. At the end of the exercise a different sound is played to indicate successful completion. The arm of the model doing the bicep curl would be highlighted in green or red, depending on whether the user was performing the exercise or not. This was done in order to demonstrate one way of providing feedback to users. In this demo the option was added to freely move the camera around the scene using the mouse. Figure 4.3 shows the initial prototype. In addition to the patient

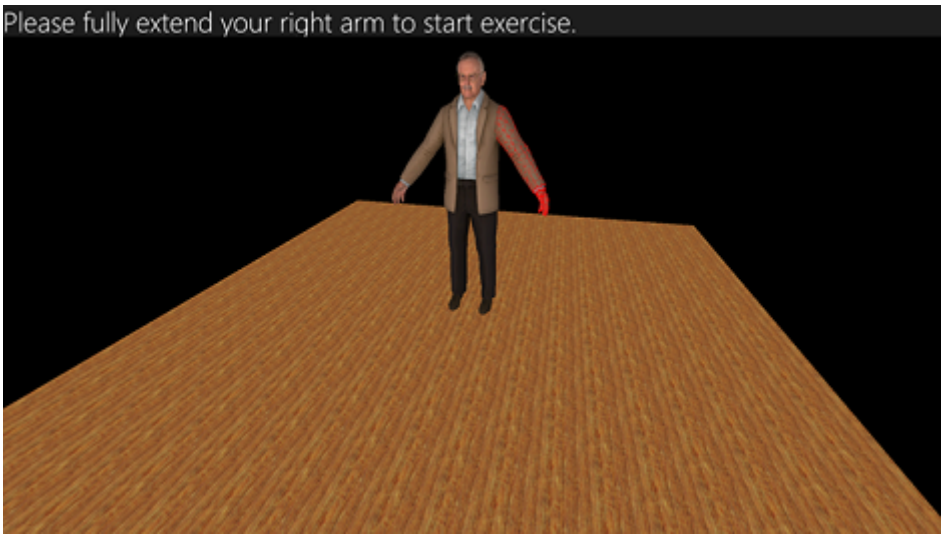

Figure 4.3: Initial prototype demonstrating a 3d avatar and limb highlighting.

application demo a paper prototype for the physiotherapist interface was developed in order to better understand the needs of the physiotherapists.

Evaluation

The prototype was evaluated in a focus group of 5 physiotherapists. The demonstration highlighted the deficiencies of using a rule-based approach. The application will count any motion as a repetition as long as the two rules defined were matched. The demonstrated prototype received mostly positive feedback. The avateering was the most commented aspect of the demo. Physiotherapists also enjoyed the provided option to freely move the camera. Results from the evaluation drew attention to the need for alternative ways to provide feedback about mistakes. Highlighting was positively received, but it does not indicate the specific mistake(s) a user was making.

Comments regarding the physiotherapist interface were also positive.

4.4.2 Second Iteration

Requirements

An outcome from the first focus group was that information was gathered about common mistakes which helped define the concept of exercise components. The previous evaluation stage also led to the realisation that highlighting can be used for indicating the presence of a mistake, but cannot indicate the nature of the mistake. This was taken into account when designing the next prototypes.

One of the goals of the second iteration stage was to design a UI for the patient application. The elements of the UI were discussed with a physiotherapist. As part of the user interface, feedback needs to be given to a patient about how well they are following their exercise programme. Inspiration was drawn from the charts presented in [23]. Finally, input methods for the application had to also be investigated. It was decided that the default Kinect interaction style will be evaluated first before any other methods are considered.

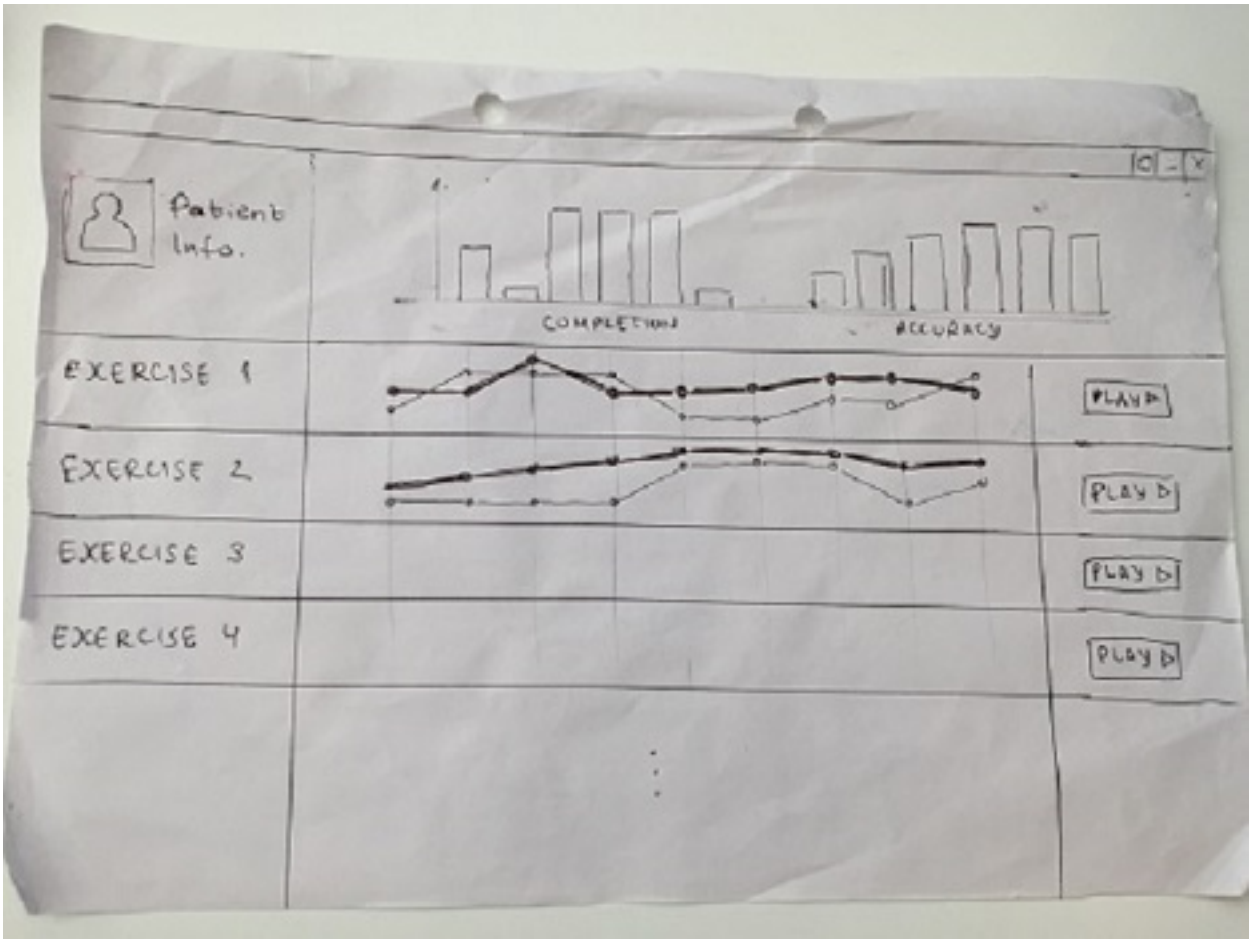

Figure 4.4: Paper prototype demonstrating the physiotherapist interface

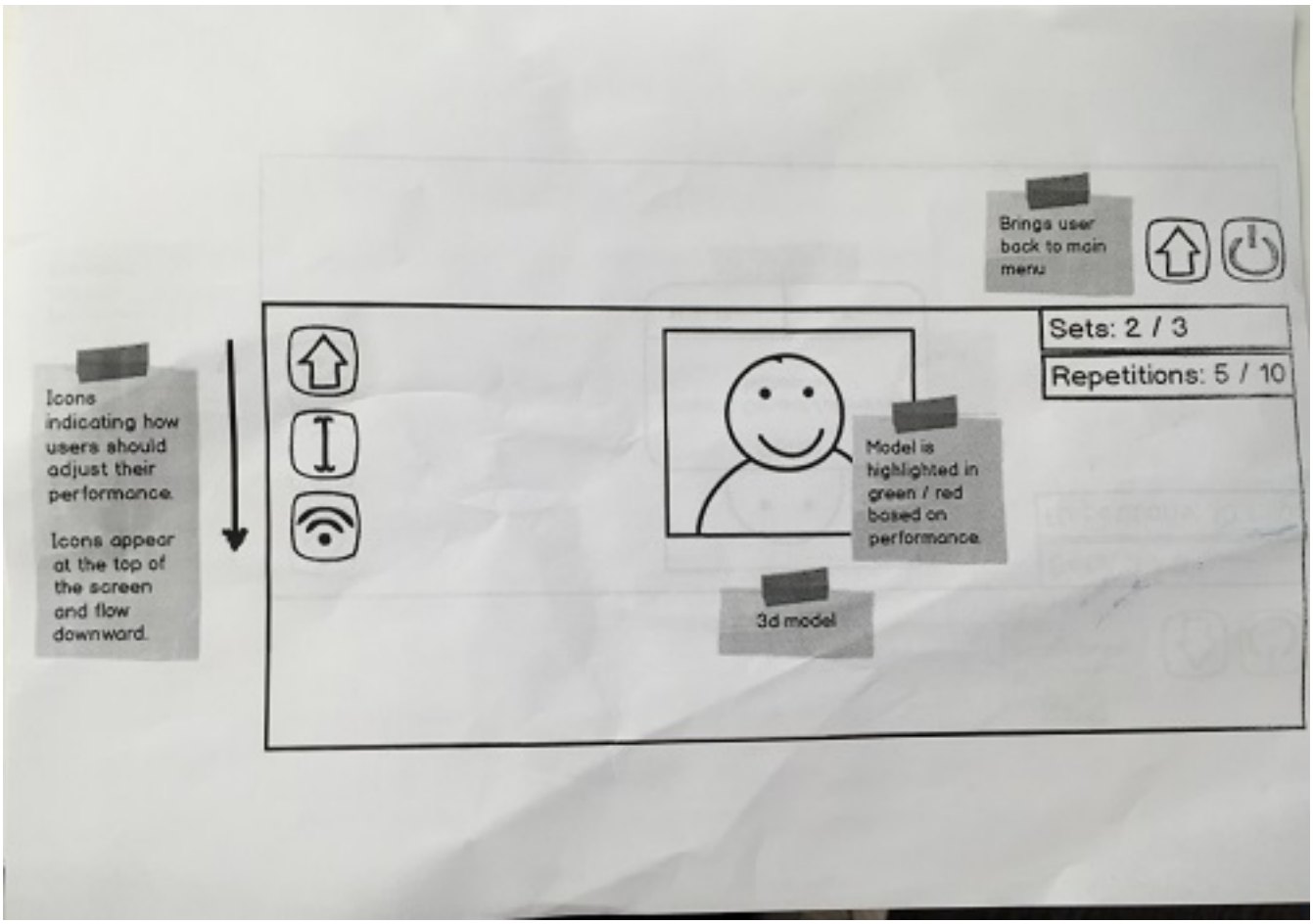

Figure 4.5: Paper prototype demonstrating exercise feedback

Prototype

Paper prototypes were developed for evaluating the user interface. A few different designs were tried, before a final one was chosen. Some non-traditional widgets were explored. To give patient some intuition about how well they are following their programme progress charts were used in the main menu and exercise pages (see navigation). A few designs were evaluated with a single physiotherapist and a final version was selected 4.5.

A second demo was developed based on the paper prototypes to demonstrate the advanced recognition capabilities of the Visual Gesture Builder algorithms, as well as the default Kinect interaction method.

In the prototypes developed a new approach was taken to giving feedback about mistakes. For each individual qualitative component an icon was chosen (Figure 4.10) which will show up in the upper left hand corner of the screen only if a patient is making the particular mistake.

Some of the features from the first prototype (such as sounds) were not included in the second prototype (due to time constraint), however even if unintentionally this produced valuable feedback when the prototype was evaluated.

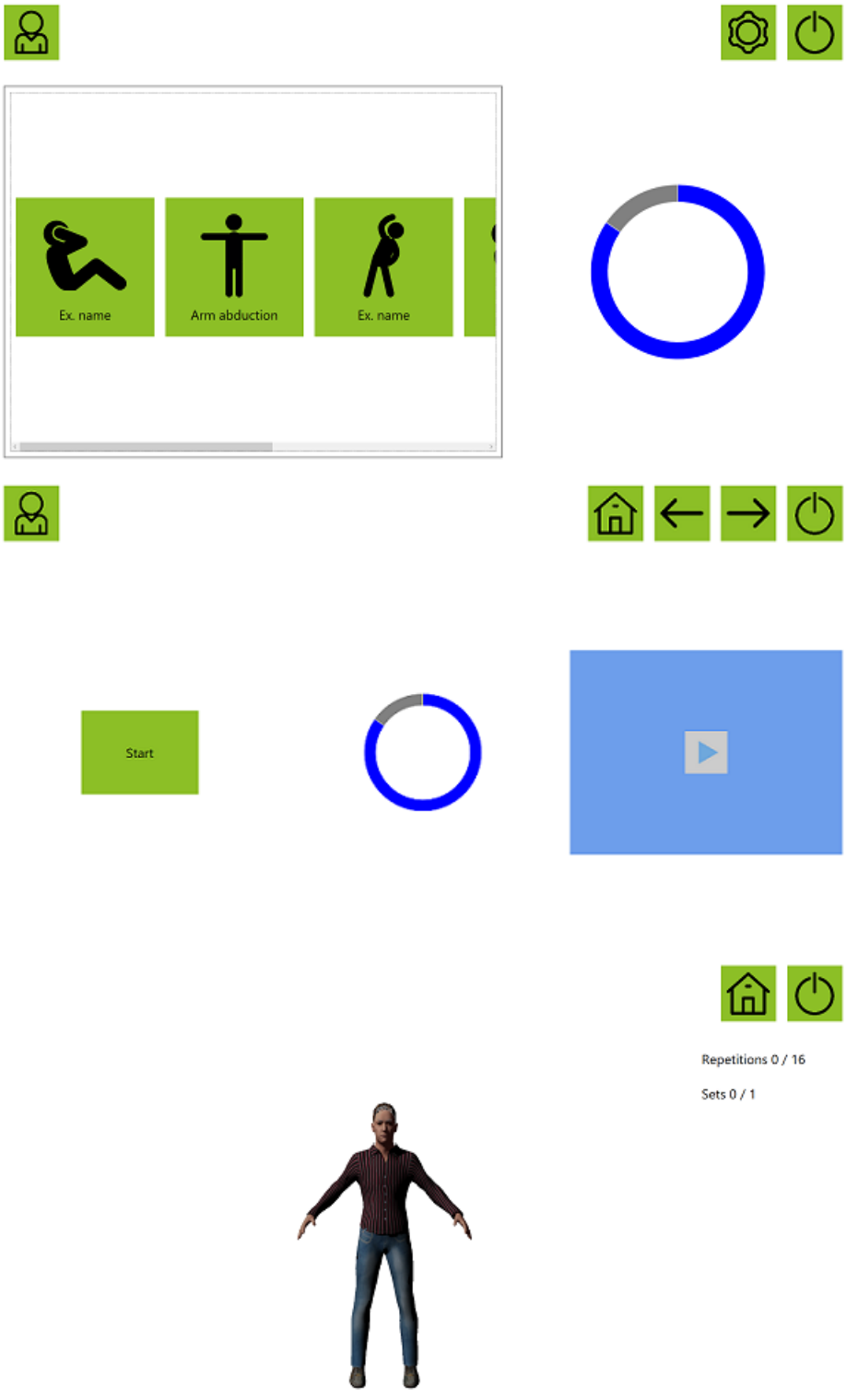

Figure 4.6: Prototype demonstrating naviation and new UI features

Evaluation

A second focus group was used to evaluate the new prototype. 3 of the participants had attended the previous focus group as well. The physiotherapists were shown both the paper prototypes and the second implementation, because some features were better presented on the paper prototypes. Some of the questions which the participants were asked related to the progress charts, because their meaning was not well understood. The feedback provided some information on how they can be improved. Feedback was less positive this time, however there were many comments regarding possible improvements and new features. One aspect participants found difficult to grasp was the Kinect interaction style.

A side effect of omitting some features from the second implementation, was that participants inquired about them, commenting that they would be useful. This validates some of the design decisions which have been made.

4.4.3 Final Iteration

Requirements

The comments from the second focus group focused on the lack of text next to component icons, which would clarify the meaning of each icon. Recommendations included making icons red and increasing the size of elements such as the repetition and set counters. Suggestions for new features included the addition of a progress bar to track progress for each repetition. Participants also requested to bring back some old features such as sounds and adding a ground for the 3D model to stand on. There was also enough feedback to warrant modifying the progress chart. Finally, a new requirement was identified - a calibration screen, which needs to check whether a user is fully visible to the sensor and whether more than one user is present.

Prototype

A third and final prototype was implemented. Sounds were added, to indicate when a rep is completed. Ground beneath the model's feed was also added back. New progress charts were designed to provide more detailed information to the user, including their best and worst exercises. Notifications, indicating consistent mistakes the user is making, were also added to the menus. The final addition was a calibration screen which pops up when the application is launched. A final version of this application should show this screen whenever the system needs calibration, not just at the very beginning. The final demo can be seen in Figure 4.7.

Evaluation

The final prototype was tested with two end users, who match the one of the main user demographics - seniors, but are not undergoing physiotherapy treatment. One participant had experience with home rehabilitation after an injury. Overall the feedback matched some of the comments from the previous evaluation - users thought the icons were too small, they weren't able to see them. Users were able to make sense of the progress charts. They liked the progress bar, saying it gave them good intuition on how to maintain correct progress throughout the repetitions. With regards to the input method used - one participant found it intuitive. The other had difficulties initially. After a few attempts, they started improving, though some difficulties and slips were still present.

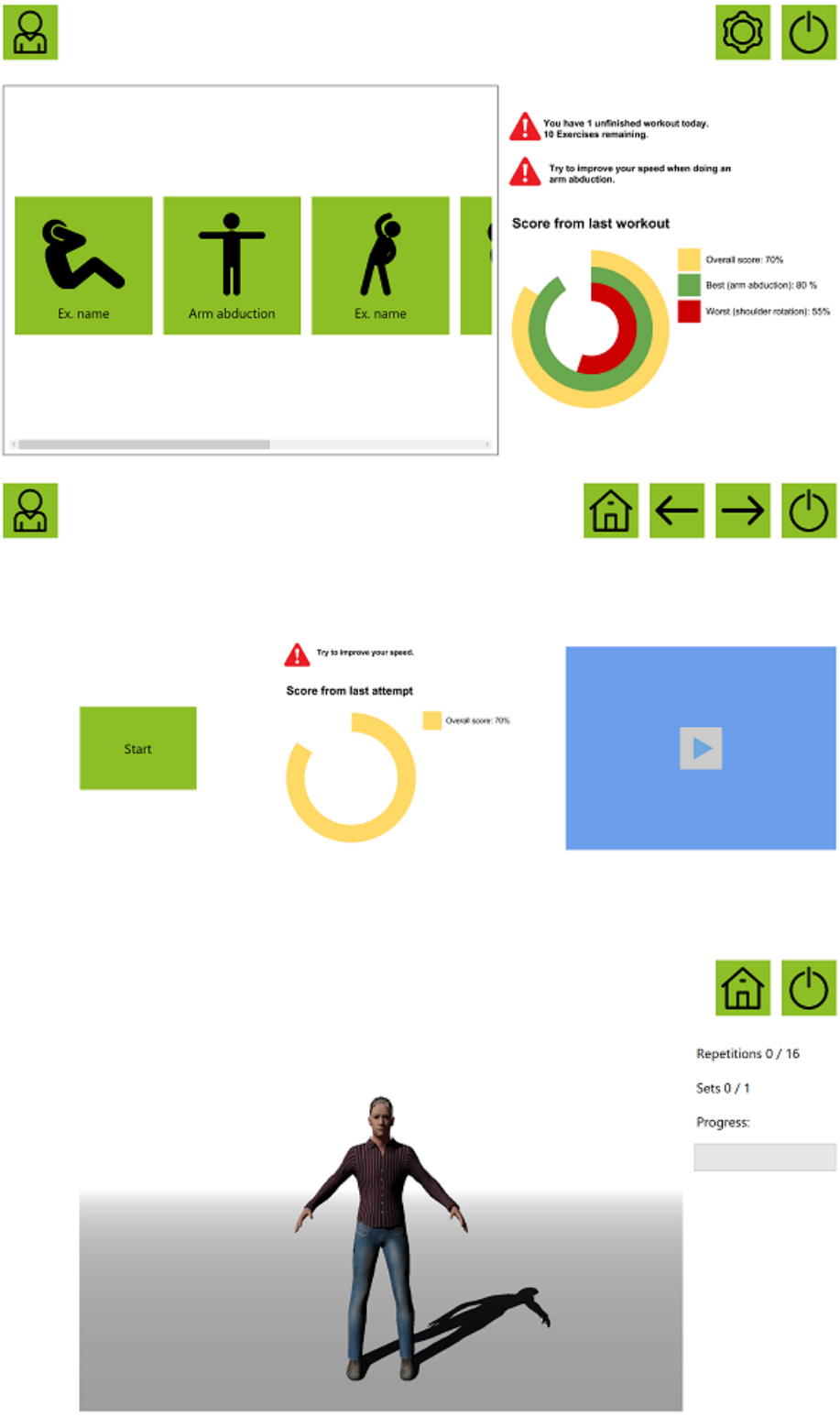

Figure 4.7: Final prototype demonstrating calibration screen and improvements from previous prototype

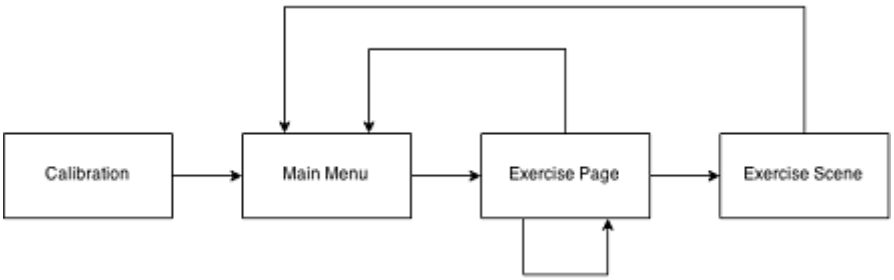

Figure 4.8: A navigation diagram of the patient application

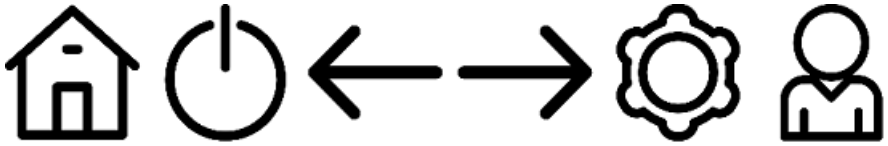

Figure 4.9: Menu icons demonstrating flat design

4.5 Design Decisions

4.5.1 Navigation

When designing the navigation for the patient application the guidelines in [32] were followed. Functionality has been grouped into pages in a meaningful way. Information regarding the entire programme is displayed in the main menu. The exercise page only contains information about the related exercise. Pages have also been used sparingly with as much of the information as possible being grouped together. Finally, consistency is provided between pages, by having a navigation bar at the top of each page. Buttons which show up in multiple pages (for example the exit button) are always displayed in the same location.

4.5.2 Aesthetics

The patient application uses flat design, in order to match the visual style of Windows 8. An increasing proportion of the elderly (one of the target demographics) [14], have used computers, therefore they should not need the metaphors relating to the real-world in order to grasp the basics of the application. The choice of design will need to be evaluated with elderly users in order to test this hypothesis. The menu icons (Figure 4.9) are an example of flat design.

4.5.3 Mistake Icons

Icons were used in order to indicate mistakes. The decision was made due to the necessity to provide UI elements large enough to be seen from a distance. Large text will take too much space on the screen. Icons on the other hand are more compact. There is a recognition - space trade-off, however. Removing text means that users will first have to learn the meaning of each icon. In an attempt to improve recognition a familiar warning sign was added to each icon, so that novice users are able to tell that something is wrong. A suggestion from the evaluation was to make these icons red, to emphasize they indicate something wrong with the performance. The icons are displayed in Figure 4.10.

Figure 4.10: Exercise component icons

4.5.4 User Input

The default Kinect interaction method was chosen as the main way for user input. It provides an interface metaphor in the form of a cursor shaped as a hand. The cursor also sticks to nearby buttons due to the reduced precision of hand navigation. Additionally, buttons in the interface were made bigger to accommodate this loss of precision. In the end, however, user evaluation showed that this input method can be difficult at times, especially for novice users.

Chapter 5

Implementation

In this chapter some of the important implementation decisions will be discussed.

5.1 Status of Implementation

The final implementation provided with this report is far from the designed system architecture. Work has only been done on the patient application. The prototype is not true to the structure presented in Figure 4.2. Many of the UI elements do not provide any functionality, they have been added only for the purposes of evaluating the user interface. The progress charts, as well as the video thumbnail have been mocked using images. The exercise page has been hard-coded to only work with the single exercise provided with the application.

5.2 Adding Exercises

An attempt has been made to simplify the task of adding a new exercise to the system. Including a new exercise involves updating the gesture database file and extending two classes - Exercise and ExerciseInstance in order to provide custom functionality for the new exercise. Updating the gesture database is done by using Visual Gesture Builder to add new gestures and export an updated database file. This step can be easily done by people who are assumed to have no programming knowledge, such as physiotherapists. Classes derived from Exercise are intended for defining static information about an exercise - such as the required number of repetitions and sets, a collection of gestures which constitute the exercise, as well as algorithms for evaluating each component. The Exercise class also contains a default method for creating an ExerciseInstance. This method can be overriden, in order to provide a custom way for creating an ExerciseInstance object.

The ExerciseInstance class contains real-time information about an exercise - such as the current value of each component. The information contained in the class is updated every time new gesture data is available. A reference to the corresponding class of type Exercise is also provided. This class only needs to be extended if the exercise requires additional real-time information, which is not provided by the existing accessor methods.

5.3 Exercise Recognition

The algorithms used for recognising an exercise are provided by the Visual Gesture Builder. The tool uses two types of indicators for detecting gestures [20]. Discrete indicators can determine whether a user is performing a gesture or not, and provide a confidence value, which indicates how certain the system is. Continuous indicators show the progress of a user for the given gesture, for example, they can tell whether a gesture is 50% complete. Algorithms used in Visual Gesture Builders include AdaBoostTrigger (based on the adaptive boosting algorithm) and RFRProgress (based on random forest regression) [3].

To integrate Visual Gesture Builder into an application a gesture database needs to be created first. This is done using a GUI tool (also called Visual Gesture Builder) which comes with the Kinect SDK. The tool lets users load data recorded with the Kinect and tag it. The tagged data is used for training the machine learning algorithms. The output of the application is a single data file that can be later used to recognise new gestures.

In order to enable gesture tracking and recognition within an application a few steps need to be completed. Firstly, instances of VisualGestureBuilderFrameSource and VisualGestureBuilderFrameReader need to be created. Then, a method needs to be registered with the VisualGestureBuilderFrameReader object. This method will be called continuously, whenever new gesture data is available. Finally, it is good practice to pause the frame reader in order to save resources, because at first there will be no bodies to track. The code to do so is shown in Listing 5.1.

```
1
2  vgbFrameSource = new VisualGestureBuilderFrameSource(
3      KinectSensor.GetDefault(),
4      0);
5  vgbFrameReader = vgbFrameSource.OpenReader();
6
7  if (vgbFrameReader != null)
8  {
9      vgbFrameReader.IsPaused = true;
10     vgbFrameReader.FrameArrived += gestureFrameArrived;
11 }
```

Listing 5.1: Initialising Visual Gesture Builder

The next step is to load the gesture database. The code which does this is shown in Listing 5.2.

```
1  using (VisualGestureBuilderDatabase db =
2      new VisualGestureBuilderDatabase(GESTURE_DB))
3  {
4      foreach (Gesture g in db.AvailableGestures)
5      {
6          if (g.Name.Equals(continuousGestureName))
7              vgbFrameSource.AddGesture(g);
8      }
9  }
```

Listing 5.2: Loading the Visual Gesture Builder database

Every Visual Gesture frame source tracks a single body at a time. Kinect assigns a unique identifier to each body that is currently being tracked. The frame source stores the identifier of the body it is currently tracking. When the frame source is created, the tracking identifier is set to 0, equivalent to no body being tracked. To start tracking a body, its identifier needs to be supplied to the Visual Gesture Builder frame source. Afterwards the frame reader needs to be unpaused, as it was initially paused. The method for starting the frame reader is shown in Listing 5.3.

```
1   public void Track(ulong id)
2  {
3      vgbFrameSource.TrackingId = id;
4      vgbFrameReader.IsPaused = (id == 0);
```

```
5 }
```

Listing 5.3: Starting the Visual Gesture Builder frame source

5.4 Exercise Components

Following the definitions given for exercise components the application recognises mistakes by continuously evaluating the 6 components. Algorithms for evaluating all but one component were developed. Not all of these algorithms feature in the final implementation, however.

5.4.1 Current State of Implementation

The latest implementation of the patient application can track repetitions and sets, as well as progress. Some attempts were made to implement control, but no working algorithm could be developed for this component.

5.4.2 Algorithms

The algorithms developed for each component are exclusively related to the arm abduction exercise, which can be modelled using 1 discrete and 1 continuous gesture. The discrete gesture corresponds to the position of the body when continuous gesture progress is 0% (i.e. the start of a repetition). It is reasonable to assume that component algorithms will vary depending on the exercise. The algorithms described in this report, however, are still useful for providing an intuition on how the output from Visual Gesture Builder can be used to recognise mistakes. An important thing to note is that Visual Gesture Builder returns float values in the range 0 to 1 for progress and confidence. These may sometimes be too accurate for the purposes of the application, so it is reasonable to add an error value, such that if the difference between two values is smaller than the error value, they should be treated as equal. When implementing the arm abduction the error value was set to 0.2, however, this value cannot be considered good with certainty as it has not been evaluated experimentally. For the sake of simplicity the error value has been omitted from the algorithm descriptions provided.

Repetitions and Sets

The algorithm for counting repetitions requires keeping track of the current and previous progress values of the continuous gestures. If the current value is 0 and the previous value is larger than 0, then a repetition has been completed and the repetition counter should be incremented. The set counter should be incremented every time the repetition counter reaches the number of repetitions required. When this occurs the repetition counter should be reset to 0.

Progress

The algorithm is similar to finding the largest value in an unordered list. Start with a variable, let's call it "max", equal to 0, whenever progress is equal to 0. Update the value of "max" accordingly until progress becomes 0 again. Return "good" if the value of "max" is 1, otherwise, return "bad". The value of max then needs to be set back to 0.

Speed

The algorithm for speed makes use of the fact that progress updates occur at a fixed time interval. Therefore, the faster a user moves, the bigger the absolute difference between two consecutive progress values will be. This difference can be considered the speed of the user. The algorithm begins by initialising an empty list at the beginning of each repetition (a new repetition begins every time progress reaches 0). Every time the progress value is updated, the absolute difference between the new value and the previous value is added to the list. At the end of a repetition, the average is calculated over all values in the list. To derive a range of ”good” average speeds, this algorithm can be applied to recordings of people doing the exercise with the correct speed.

Posture

Posture is the only component which uses the discrete gesture. It was observed, that with the arm abduction exercise Visual Gesture Builder will keep returning progress values even if the user's posture is incorrect. The confidence value for both discrete gesture, however, will be low. The algorithm for evaluating posture takes advantage of this observation. At the end of each repetition the confidence value is evaluated. If it falls below a certain threshold (which is determined empirically), then the posture is incorrect.

Control

Designing an algorithm for evaluating control is not a trivial task. Based on the definition for control, the intuition is that the component can be represented by measuring the amount of variation in the set of speeds for a given repetition. The standard deviation was used as a measure of how close the speeds are to each other. The smaller the value, the better the control. Calculating the standard deviation over a set of example repetitions could, in theory, yield a value, which could be used to categorise a repetition as good or bad. The algorithm was evaluated with a set of 25 example repetitions (15 good and 10 bad) and the results showed that bad repetitions can have values both smaller and larger than good repetitions, making it difficult to determine a cut-off value.

Further analysis was done by plotting good and bad examples as time-series bar charts (Figure 5.1). An observation was made, that good attempts appear similar to a Gaussian function, whereas bad attempts look seemingly random. A possible algorithm would be to fit a Gaussian function to the progress data for a repetition and then compute the similarity between the function and the actual data. A possible similarity value is the mean squared error. This approach has not been tested on actual data, so its feasibility is unknown.

One final conclusion was reached by observing recorded data - the set of progress values for a good repetition is much smaller than the set for a bad one. This alludes to the possibility of there being a collection of features which can determine, whether a repetition is good or bad. In this case, using a machine learning algorithm may turn out to be a viable option.

5.5 Avateering

Avateering is a term, which appears in a Kinect demo released by Microsoft [2]. It is presumably a portmanteu of the words avatar and steering, and describes the activity of controlling a 3d avatar with one's body. The aforementioned demo was developed for the Kinect V1 and uses the joint orientations provided by the SDK to manipulate the 3d mesh. At the time of implementation, the Kinect SDK V2 was still in preview stage and there were some inaccuracies with joint orientation values. An alternative was to calculate the correct values using joint position data. An article found online [8] mentions an avateering demo for the Kinect V2. The author of the

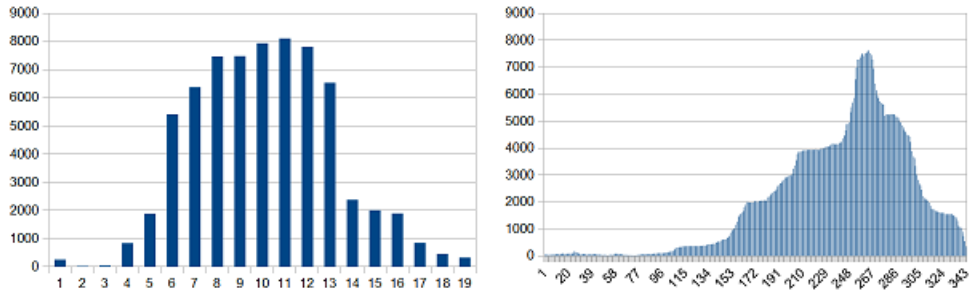

Figure 5.1: Comparison between ”good” control (left) and ”bad” control (right)

article was contacted with a request to provide a demo. The code he supplied was provided with a free software license. The author's only request was to include a link to his site in this report. The algorithm used in the patient application is based on ideas from his demo.

In computer graphics it is common for 3D meshes to have an attached skeleton, which consists of bones. Each bone is given a unique name and is responsible for controlling parts of the mesh. Bones can be rotated programmatically, which results in the mesh being animated. Each bone consists of a head and a tail. Bones can be attached to each other, forming a hierarchy, with the head part being attached to a parent and the tail part being attached to a child.

In order to implement avateering a map structure needs to be used in order to associate Kinect joints to bones. An example is shown in Listing 5.4.

```
1 mapping = new Dictionary<string, JointPair>();
2 mapping.Add("upper_arm.L", new JointPair(JointType.ShoulderRight, JointType.ElbowRight));
3 mapping.Add("forearm.L", new JointPair(JointType.ElbowRight, JointType.WristRight));
4 mapping.Add("upper_arm.R", new JointPair(JointType.ShoulderLeft, JointType.ElbowLeft));
5 mapping.Add("forearm.R", new JointPair(JointType.ElbowLeft, JointType.WristLeft));
```

Listing 5.4: Mapping Kinect joints to mesh bones

The next step is to apply the latest joint data to the bones every time a new frame is about to be rendered. Vector algebra is used in the process. A direction vector is calculated for each bone, based on the associated Kinect joints. The rotation between this vector and the bones current direction vector is applied to the bone. A portion of the implementation is shown in Listing 5.5. The full avateering implementation can be found in Appendix C.

```
1 foreach (Bone b in skeleton.GetBoneIterator())
2 {
3     if (mapping.ContainsKey(b.Name))
4     {
5         Vector3 current = getBoneLocalDirectionVector(b);
6         Vector3 target = getTargetLocalDirectionVector(b);
7         b.Rotate(current.GetRotationTo(target));
8     }
9 }
```

Listing 5.5: Mapping Kinect joints to mesh bones

5.6 Tools and Languages

5.6.1 Languages and Frameworks

The language used in the implementation was C#. For developing the user interface the Windows Presentation Foundation (WPF) [22] framework was used. Graphics were provided by Mogre [12] - a C# wrapper for the Ogre [13] graphics library. The Microsoft Kinect SDK [7] was used to get data from the Kinect sensor.

The initial demo was developed using different tools and languages. These were scrapped due to compatibility isses. The first demo was written in JavaScript, using the Three.js [18] graphics library.

5.6.2 Development

The application was developed for Windows 8 due to the requirements of the Kinect SDK. The integrated development environment was Visual Studio 2013.

5.6.3 Additional Applications

Additional applications used were Kinect Studio, Visual Gesture Builder and Make Human. Kinect Studio is a tool for recording videos from the Kinect sensor, which come with the SDK. It is used to create recordings which are then analysed in Visual Gesture Builder. Make Human [9] is a free open source application for generating 3d human meshes with skeletal data.

Chapter 6

Evaluation

Evaluation was a crucial part of the iterative design process. This chapter reviews the evaluation methodologies used throughout this project and discusses in detail the results produced from evaluation. The implementation was mainly evaluated within two focus groups consisting of 5 physiotherapists each. A final implementation was evaluated with two elderly users. Heuristics were also applied to the design of the final prototype.

6.1 Focus Groups

Two focus groups were conducted with physiotherapists with different specialisations, some of them having experience with amputees, while other with the elderly. Input from physiotherapists is very valuable, because they are the subject experts, who have knowledge on how to assess exercise correctness and what the common mistakes are. They are also experienced in providing feedback to patients and know how to communicate mistakes to their patients. Physiotherapists are also aware of the common questions asked by patients, which is useful when determining what information should be displayed on the user interface.

Some of the participants took part in both focus groups, while others only attended one. This was useful when conducting the second focus group, because there was a good balance of people who had previous knowledge about the project and people who were just learning about it. This encouraged the two parties to share their knowledge, getting them to discuss the project amongst each other.

Both focus groups were conducted in a similar fashion. In the beginning, participants were given a brief introduction about the project. During the second focus group paper prototypes were used as a visual aid. Afterwards, participants were asked some questions related to project requirements which were still poorly understood. Following these questions they were given time to familiarise themselves with existing prototypes. They were encouraged to provide their thoughts and opinions in the process, by being asked questions about different features presented in the prototypes. Finally, some questions asked before the demonstration were revisited, in order to see whether gaining knowledge about the system has changed participants' opinions in some way.

The outcomes from the two focus groups were slightly different. The first focus group put more emphasis on understanding the requirements for the project and the target users. It also served as a first-time demonstration of the capabilities of the Microsoft Kinect. Feedback from the first focus group was mostly positive. It centered around the capabilities of the system to track an exercise and the decision to use avateering.

Much more feedback was provided during the second focus group. Participants expressed mixed opinions about some features of the user interface. A lot of changes were suggested for existing elements of the application

and additionally a few new features were proposed. Comments were also made regarding the input method of the application.

Overall, the two focus groups generated a lot of feedback and greatly helped in the understanding of some of the requirements. However, it is worth noting that feedback from physiotherapists is not enough, because they are not the end users of the patient application. In the future the application will have to be tested with patients. Given that no patients were involved in the evaluation, there is a possibility that some of the results reported in this project may not be entirely accurate and some of the stated requirements may need revision.

6.2 End User Evaluation

The final evaluation with two users occurred in a similar way to the two focus groups. The participants were given a brief introduction to the system and then they were encouraged to try it for themselves.

Feedback regarding the interaction method was mixed. One participant found it difficult and unintuitive. They did improve their accuracy with time, however. The other participant found it much easier to navigate using hand movements. So far the overall feedback regarding the Kinect input method has been mixed.

The two participants were asked to have a look at the updated progress charts, and were then asked whether they understand the meaning. Both agreed that the information related by the progress charts, as well as the notifications, was clear.

6.3 Recommended Features

Some of the features suggested by physiotherapists and end users include:

- Adding colour to the component warning icons.
- Add an axis starting from the origin of the scene passing straight through the model. This way users can observe their posture.
- Add the option to change the avatar's gender.
- Add warning text telling users what their mistake is. Icons don't seem to be enough.

6.4 Summary

Summarising the results, the application received enough positive feedback to be considered feasible at this stage. The main issues were with the interaction method. Physiotherapists were mostly impressed by the avateering demo, so it seems like something that should be evaluated with end users to get their opinion. The two end users did not express too much excitement regarding the 3D model.

Chapter 7

Conclusion

7.1 Insights

This project attempted to look at a broad range of topics in the field of home rehabilitation with the Kinect. The main goal was to examine the feasibility of a distributed physiotherapy system which extends from the patients' homes all the way to the clinic.

Some of the aspects this project tried to look into, which have not been thoroughly examined before:

- Only one other project [24] examines the feasibility of an entire distributed system. The difference between this project and the other one have been highlighted.
- No physiotherapy related project discusses any input methods.
- Few if any projects examine the user interface which accompanies a physiotherapy application.
- No project has used the Kinect V2 and its SDK so far. The Visual Gesture Builder algorithms have not been tried for rehabilitation exercises.
- This project proposes a new model for finding mistakes in an exercise (using components).

7.2 Challenges

The main challenge of the project came from the implementation. Computer graphics is something I have never done before and the learning curve was steep. Additionally, avateering had to be implemented from examples where code was poorly documented. This made the challenge even bigger.

Another challenge was finding the right set of tools and languages which will be compatible with each other. The initial prototype was written using an entirely different language and graphics library. These were scrapped because Visual Gesture Builder could not be integrated with the language. Finding a suitable C# graphics library was challenging.

7.3 Suggestions for Future Work

7.3.1 Exercise Recognition

Algorithms for evaluating most components were designed, however none have been properly evaluated. An experiment needs to be conducted in other to check whether the proposed algorithms are actually useful. Machine Learning may have to be used when evaluating control, if the other suggestions prove to be impractical.

7.3.2 Physiotherapist Interface

Feedback for the initial paper prototype of the physiotherapist interface was positive. Additional iteration cycles are necessary in order to refine the design and create a more high-fidelity prototype.

Additional pages of the physiotherapist interface need to be designed.

7.3.3 Evaluation with End Users

However useful, the two focus groups are not enough to evaluate the patient application concept. Eventually a prototype needs to be tested with actual patients, in order to determine whether the decisions made are really valid.

7.3.4 Interaction

Alternative interaction methods need to be evaluated. User feedback and behaviour indicated there are difficulties with the Kinect default input method. Interaction methods to be evaluated may include - remote controls, game controllers and speech.

Appendices

Appendix A

Source Code Repository

The source code for the project is stored in the following Github repository:

https://github.com/vshulev/kinect-physiotherapy.

The final prototype is located in branch ”dev”. The first prototype is located in branch ”3d-model-prototype”.

Appendix B

Running the Prototypes

To run the prototypes simply import them into Visual Studio 2013 by clicking on the ".sln" file in the root directory. The project uses NuGet for dependency management, so all dependencies should be resolved when the project is first built (from within Visual Studio 2013).

Appendix C

Avateering Implementation

This appendix provides a listing of all relevant avateering methods.

```
1  private void createMapping()
2  {
3      mapping = new Dictionary<string, JointPair>();
4      mapping.Add("upper_arm.L", new JointPair(JointType.ShoulderRight, JointType.ElbowRight));
5      mapping.Add("forearm.L", new JointPair(JointType.ElbowRight, JointType.WristRight));
6      mapping.Add("upper_arm.R", new JointPair(JointType.ShoulderLeft, JointType.ElbowLeft));
7      mapping.Add("forearm.R", new JointPair(JointType.ElbowLeft, JointType.WristLeft));
8  }
```

Listing C.1: Creating a mapping between bones and Kinect joints

```
1  private void createMapping()
2  {
3      mapping = new Dictionary<string, JointPair>();
4      mapping.Add("upper_arm.L", new JointPair(JointType.ShoulderRight, JointType.ElbowRight));
5      mapping.Add("forearm.L", new JointPair(JointType.ElbowRight, JointType.WristRight));
6      mapping.Add("upper_arm.R", new JointPair(JointType.ShoulderLeft, JointType.ElbowLeft));
7      mapping.Add("forearm.R", new JointPair(JointType.ElbowLeft, JointType.WristLeft));
8  }
```

Listing C.2: Creating a mapping between bones and Kinect joints

```
1  public void Update()
2  {
3      if (joints == null)
4          return;
5      foreach (Bone b in skeleton.GetBoneIterator())
6      {
7          if (mapping.ContainsKey(b.Name))
8          {
9              Vector3 current = getBoneLocalDirectionVector(b);
10             Vector3 target = getTargetLocalDirectionVector(b);
11             b.Rotate(current.GetRotationTo(target));
12         }
13     }
14 }
```

Listing C.3: Method called once before each frame is rendered

```
1  private void initializeSkeleton()
2  {
3      skeleton = entity.Skeleton;
4      foreach (Bone b in skeleton.GetBoneIterator())
5      {
6          b.SetManuallyControlled(true);
7          b.InheritOrientation = true;
8          b.InheritScale = true;
```

```
9       }
10  }
```

Listing C.4: Initialise mesh skeleton to enable animation

```
1       private Vector3 getTargetLocalDirectionVector(Bone b)
2       {
3           JointPair pair = mapping[b.Name];
4           Vector3 worldDirection = getWorldDirectionVector(pair.Start, pair.End);
5           return translateWorldToLocalDirectionVector(b, worldDirection);
6       }
7
8       private Vector3 translateWorldToLocalDirectionVector(Bone b, Vector3 v)
9       {
10          Vector3 translation = b.ConvertLocalToWorldPosition(Vector3.ZERO);
11          translation = v + translation;
12          Vector3 targetDirectionLocal = b.ConvertWorldToLocalPosition(translation);
13          targetDirectionLocal.Normalise();
14          return targetDirectionLocal;
15      }
16
17      private Vector3 getWorldDirectionVector(JointType startJoint, JointType endJoint)
18      {
19          return getWorldDirectionVector(convertJointToVector(startJoint), convertJointToVector(endJoint
                ));
20      }
21
22      private Vector3 getWorldDirectionVector(Vector3 start, Vector3 end)
23      {
24          Vector3 dir = end - start;
25          dir.Normalise();
26          return dir;
27      }
28
29      private Vector3 convertJointToVector(JointType joint)
30      {
31          if (joints == null)
32              return Vector3.ZERO;
33          return new Vector3(joints[joint].Position.X, joints[joint].Position.Y, -joints[joint].Position
                .Z);
34      }
35  }
```

Listing C.5: Helper methods

Appendix D

Using the Visual Gesture Builder Application

A 2 part series on how to work with Visual Gesture Builder is provided here:

http://channel9.msdn.com/coding4fun/kinect/Custom-Gestures-Kinect-for-Windows-v2-and-the-Visual-Gesture-Builder.

Appendix E

Notes Taken After Watching Video From First Focus Group

Q: How do you assess if exercise is being done properly?

A: Eyeballing patient to assess correctness. Ask someone to move just arm or shoulder, but instead they move their whole body.

Q: Does this mean most exercises concentrate on certain body parts instead of the whole body?

A: Target a particular muscle or involve whole body.

Q: Does this mean there are two different types of exercises?

A: Isolate certain muscles. Functional see certain movements e.g. sit to stand or single leg stand. Start by breaking it down into single movements then moving to functional.

Q: Tell us more about common exercises that patients do? Isolation and functional as well.

A: Movement around butt area. Difficult to isolate to give them feedback. Exercise: lying on the ground lifting one leg. Difficult for patients to isolate just that muscle, may move other parts of body.

Demonstrate to patients first how to do exercise then let them copy you OR put your hand on where you want them to work.

Problem is we give people exercises and ask them to do them on their own. Physio will know if patient has done exercise or no by observing change when a patient visits there will be no difference at all or fantastic difference.

Q: How often do patients have to report on progress? How often do they go see physio?

A: Depends on type of physio and type of problem patients have. Ranges from daily to monthly. Depends on how much change you expect to see.

Q: Improvement would be more reps or longer range of movement?

A: Could be these two as well as pattern endurance of the muscle.

Q: How do you measure that?

A: Physical resistance. Pushing against something or trying to lift against gravity maybe an object or just limb itself.

Grade it with the Oxford scale 1 to 5.

Speed of movement is sometimes is also important sign patient is getting better. Speed and quality as well control, fluidity, not shaking, nice and smooth continuous movement rather than jerky.

Q: What about other exercises?

A: 8:53 Stand and hold on to a work surface. Raise a single leg. Move it sideways or backwards. 10:50 Raising arm. 11:48

Q: Do you do home visits?

A: There is a team that does that.

Q: In the clinic do patients do exercises under physio supervision or are they told to do an exercise before the physio moves on to the next patient?

A: A Kinect could be useful in the clinic.

Q: Average schedule would involve traveling between places or being based at the clinic?

A: Go about from ward setting and gym setting. Kinect would be useful because a lot of physios have groups/classes.

Q: Would a group of patients all be doing the same exercises or exercises related to the same groups?

A:

Q: Give us an exercise that people would commonly do badly.

A: 15:51 Squats.

Speed sometimes slow is good, sometimes fast is good.

Q: One model of how an exercise is to be assessed might not work for all exercises?

A: Exercises may also be modified according to patient.

Q: Would it be beneficial for you to see what patients are doing?

A: May make patients comply with exercises if patients know they are being watched. Time spent with physio is not enough, patients need to follow a routine at home.

Physios give patients a set of exercises to do at home, then tell them when to come visit them again.

If patients have too many things to do, they may lose interest.

Exercises are usually written down using a tool PhysioTools, a very basic computer program. Sometimes if you give patients to much to do, they don't bother.

Q: Is PhysioTools something patients have access to?

A: Exercises are selected from there and then printed out. Physios work with PhysioTools.

Q: Would you spend time on viewing individual exercises, would you be more interested in aggregated data of overall patient progress?

A: Would view individual exercises. Not so much about progress more about preventing them from getting problems later on.

View individual exercises as a diagnostic tool only view when you observe patients are not making progress, to determine why they are not making progress.

Useful option would be to pick videos to view.

02:40 Pointing at prototype.

Q: Is this information overload or does it look like the thing you would like to have?

A: Don't think it's an information overload. Doubt if patients would adhere to plan, however, you'd know if patient had done exercises.

Can tell you which exercises are agreeable to patients. If everybody not doing a particular exercise (well) might not be agreeable or taught properly.

We've got completion and accuracy, but we've not got speed, compensatory movements. Accuracy sounds fine, but could also be quality. It would however be measured differently for different exercises.

Q: Accuracy is a composite value that may contain things such as fluidity, speed, etc. Should these be displayed separately as well?

A: If you were to press play, you'd be able to see for yourself.

Q: So instead of trying to analyze data too much, just have the play button available? We could thoroughly analyze data to try and determine what's going wrong or provide a play button for you to assess it.

A: Play button would be more useful. Would you have the ability to display an axis when you've pressed play?

A physio tool for amputees that measures limb angles.

Be able do get a break down of the data, would be quite useful.

Q: What would this breakdown look like?

A: Click on accuracy and get it broken down into further values e.g. speed, compensatory movements, power, fluidity, etc.

Q: Is this the same kind of thing you would want to see for a patients, who's been doing their exercises at the clinic, versus a weekly summary of patient activity at home?

A: Time thing shouldn't really matter, it's whether they're doing it correctly.

Q: What would you look the data on? Phone, computer?

A: Computer yeah, not allowed to on phone needs to be secure, password protected. Tends to be laptops. Tablets?

Q: Do you think this data would be useful to patients as well?

A: More savvy ones (joke). For patient something like that along with the real time stuff.

Q: Would it also be useful to rotate around the patient when viewing how they're doing a certain exercise?

A: It would be important for hips and shoulder.

If patients can see the center of their balance, they can adjust accordingly.

It would be useful for patients to have a guiding line that can tell them how to adjust their bodies.

Q: Does this prototype look like something a patient might make sense of?

A: Deffinitely. Especially the color thing.

Parallel bars?

Appendix F

Notes Taken After Watching Video From Second Focus Group

Q: Score thing? Doest it make sense to give them some score of what they've been doing? What is it? Is it just number of repetitions you've done or is it the quality of the action?

A: ¡no answer yet¿

A: some people might be happy about being represented as another person (joke)

Q: Does this description of fluidity make sense (i.e. variations in speed)? Is there more to it?

A: Yep, it's ok.

A: Would it be possible insted of having these icons looking like that, to actually have text?

Q: Can you give as a short string that would capture the meaning? Putting too much text will obstruct view? Is fluidity a good name?

A: More appropriate would be control. You would say to the patient "control your movement". Steady speed. Maybe not text even, but something that's not as small (referring to icons). Even red color. Can have the same warning symbol, but also text e.g. "Go slow!" or "Keep control!". Quite like the fluidity one, but the progress one is not the best. Add sounds e.g. a small beep when an icon pops up.

Q: Can even have beep for every repetition.

A: Or when you get to the top (max progress) get a bing (x2 suggestions for this). Maybe use Machine Learning to learn patient speed, so that it doesn't warn them about speed all the time.

Q: Does it matter at what speed patients do the exercise? Speed is not featured in the demo.

A: Yes.

Q: Is it too fast AND to slow, or just too fast?

A: Sometimes it's too slow. Depends on the exercise, for some you can do too slow.

Q: Would you want a warning if someone was doing the exercise really fast?

A: Yes. You don't want patients to just drop their arm while lowering it.

Q: You mentioned icons being too small. Where would you place them on the screen?

A: Above head or below feet? Model is also quite small, it'd be better if he was bigger.

Q: If the exercise does not involve the whole body should the whole body be displayed on screen?

A: Yes (a few people). I think those icons would be fine at the side, if only they were bigger.

Q: Are icons displayed long enough? Should they be displayed longer? If you, however do the next repetition correctly and there's still a message, then wouldn't that be confusing.

A: Makes sense. Could you have positive feedback as well green icon saying well done, if you do a succession of succesful repetitions? To reinforce they do it right. Q: Do you think any highlighting on the model would be useful? Last time you can put red/green shading on the model to indicate errors.

A: Doesn't really matter whether you have color on the model or not... I quite like the last time when the arm would be highlighted. *Some of the physios explaining the previous demo to the ones that haven't seen it*.

Q: If it's red it doesn't tell you what's wrong, just that something's wrong. Could be a combination of both.

A: May be good, because if you enable more parts of the body, the problem area of the body could be highlighted. Can you have a progress bar 0 100% that goes up while doing the rep. I think that's a good idea to have on the other side of the screen.

Q: How would it fill up where's 50% and where's 100%?

A: Fill up all the way when you raise you arm sufficiently, and then go back down.

Q: If you can see a bar, then would you need a picture of a human there?

A: It's extra feedback. I think it's still good because motor theory watching your movement good or self-improvement.

Q: Would you like the person to stand on something, have a background?

A: Last time there was a floor surface, looked more 3d.

Q: What about putting the avatar in a more realistic setting (e.g. a room)?

A: Blank space is a bit off-putting. Better to have some floor. No sofas and stuff though. Maybe a carpet? Just a floor will be fine, no furniture. Thinking of the Wii background, like a room but empty.

Comment: Already got a little bit of workout there.

A: Can posture be implemented?

Q: Yes

A: Make the repetition and sets text bigger. Will it tell you if you're doing the wrong exercise?

Q: No. What do you think of this style of control?

A: I find that difficult.

Q: Standard Kinect style, but not necessarily the easiest thing to do.

A: Older people might find this kind of technology difficult. Exercise selection field quite small could be made wider. It would be easier to move. We also talked about voice activation, didn't we?

Q: Yes, it's possible. It promises more than it delivers. What about using the TV remote?

A: Make icons bigger.

Q: Ratings what do you think would be useful for patients to know? Last exercise score.

A: You can have a score for "have you done your exercise?" Have they been missing an exercise? Have they been missing the sets? Also, how well they do overall quality.

Q: What would the quality represent? Over all exercises? Presumably you're prescribing them multiple exercises rather than just one type. Is is a measure across all of those? What if you do one really well and one really badly?

A: Should be an average percentage for overall quality. Say you have 10 exercises on exercise (not a mistake) 1 and you do 5 of them well you've got 50%. Could you do an average?

Q: We could, but do you think this is something that patients will find useful?

A: Could tell which one they did best and which one they did worst? So they know what they need to work on. If it's one conglomerate instead of individual scores for each component, that wouldn't mean much to patients. You could have a conglomerate percentage and then tap on that to get a summary of different exercises if you wanted to. You can give score as stars.

Q: Is it typical thing you gotta do those arm raises 10 lots 3 times a day every day for 2 weeks. Is that a normal type of thing? So daily goals?

A: We need number of repetitions, number of sets, number of times a day, number of days a week, number of weeks.

Q: You might need that, but would the patient need that as well?

A: Have a bar chart or a line graph or whatever saying whether you've done them each day and that can change over the week.

Bibliography

[1] Amazon web services. http://aws.amazon.com/.

[2] Avateering c# sample. https://msdn.microsoft.com/en-us/library/jj131041.aspx.

[3] Detection technologies. https://msdn.microsoft.com/en-us/library/dn785523.aspx.

[4] Google cloud platform. https://cloud.google.com/.

[5] Jintronix. http://www.jintronix.com/.

[6] Kendo ui. http://demos.telerik.com/kendo-ui/spa/index.

[7] Kinect for windows sdk 2.0. http://www.microsoft.com/en-gb/download/details.aspx?id=44561.

[8] Kinect v2 with ms-sdk. http://rfilkov.com/2014/08/01/kinect-v2-with-ms-sdk/.

[9] Make human. http://www.makehuman.org/.

[10] Microsoft azure. http://azure.microsoft.com/en-gb/pricing/free-trial/?WT.srch=1&WT.mc_id=Paid_azure_Free_Trial+A

[11] Microsoft kinect. http://www.microsoft.com/en-us/kinectforwindows/meetkinect/default.aspx.

[12] Mogre. http://www.ogre3d.org/tikiwiki/MOGRE.

[13] Ogre. http://www.ogre3d.org/.

[14] Older adults and technology use. http://www.pewinternet.org/2014/04/03/older-adults-and-technology-use/.

[15] Physiotools. https://ptfi.physiotoolsonline.com/?PageId=Home&Redirect.

[16] Reflexion health. http://reflexionhealth.com/.

[17] Stroke - recovery. http://www.nhs.uk/Conditions/Stroke/Pages/recovery.aspx/.

[18] three.js. http://threejs.org/.

[19] Virtual rehab. http://www.virtualrehab.info/.

[20] Visual gesture builder: A data-driven solution to gesture detection. https://onedrive.live.com/view.aspx?resid=1A0C78068E0550B5!77743&app=WordPdf.

[21] Wii fit. http://wiifit.com/.

[22] Windows presentation foundation. https://msdn.microsoft.com/en-us/library/ms754130%28v=vs.110

[23] Mobolaji Ayoade and Lynne Baillie. A novel knee rehabilitation system for the home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 2521–2530, New York, NY, USA, 2014. ACM.

[24] F. Cary, O. Postolache, and P. Silva Girao. Kinect based system and artificial neural networks classifiers for physiotherapy assessment. In *Medical Measurements and Applications (MeMeA), 2014 IEEE International Symposium on*, pages 1–6, June 2014.

[25] F.C. Cary, O. Postolache, and P.M. Giro. Kinect based system and serious game motivating approach for physiotherapy assessment and remote session monitoring. In *International Conf. on Sensing Technology - ICST*, volume 1, pages 1–5, September 2014.

[26] Brian X. Chen. Playing at no cost, right into the hands of mobile game makers. *The New York Times*, 2012.

[27] Alan Cooper. *The Inmates Are Running the Asylum*. Macmillan Publishing Co., Inc., Indianapolis, IN, USA, 1999.

[28] J. Couto Soares, A. Vieira, and J. Gabriel. Assisted living: Home physiotherapy demo. In *Experiment@ International Conference (exp.at'13), 2013 2nd*, pages 162–163, Sept 2013.

[29] A. Da Gama, T. Chaves, L. Figueiredo, and V. Teichrieb. Poster: Improving motor rehabilitation process through a natural interaction based system using kinect sensor. In *3D User Interfaces (3DUI), 2012 IEEE Symposium on*, pages 145–146, March 2012.

[30] Oonagh Giggins, Kevin Sweeney, and Brian Caulfield. Rehabilitation exercise assessment using inertial sensors: a cross-sectional analytical study. *Journal of NeuroEngineering and Rehabilitation*, 11(1):158, 2014.

[31] Tomasz Hachaj and MarekR. Ogiela. Rule-based approach to recognizing human body poses and gestures in real time. *Multimedia Systems*, 20(1):81–99, 2014.

[32] Dr. Charles Kreitzberg and Ambrose Little. Strategies for designing application navigation. https://msdn.microsoft.com/en-us/magazine/dd458810.aspx.

[33] B. Lange, Chien-Yen Chang, E. Suma, B. Newman, A.S. Rizzo, and M. Bolas. Development and evaluation of low cost game-based balance rehabilitation tool using the microsoft kinect sensor. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 1831–1834, Aug 2011.

[34] Aarthi Ravi. Automatic gesture recognition and tracking system for physiotherapy. Master's thesis, EECS Department, University of California, Berkeley, May 2013.

[35] Kyle Rector, Cynthia L. Bennett, and Julie A. Kientz. Eyes-free yoga: An exergame using depth cameras for blind & low vision exercise. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '13, pages 12:1–12:8, New York, NY, USA, 2013. ACM.

[36] Stephen Uzor, Lynne Baillie, Dawn Skelton, and Fiona Fairlie. Identifying barriers to effective user interaction with rehabilitation tools in the home. In Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler, editors, *Human-Computer Interaction INTERACT 2011*, volume 6947 of *Lecture Notes in Computer Science*, pages 36–43. Springer Berlin Heidelberg, 2011.

[37] Wenbing Zhao, Hai Feng, R. Lun, D.D. Espy, and M.A. Reinthal. A kinect-based rehabilitation exercise monitoring and guidance system. In *Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on*, pages 762–765, June 2014.