HTBLuVA Wiener Neustadt
Höhere Lehranstalt für Informatik
Ausbildungsschwerpunkt

DIPLOMARBEIT

InMoov as a Physiotherapist

Ausgeführt im Schuljahr 2016/17 von:

Christoph Käferle	5AHIF
Max Hoffmann	5AHIF

Betreuer / Betreuerin:

Dr. Michael Stifter

Wiener Neustadt, am 27th March, 2017

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

Wiener Neustadt, am 27th March, 2017

Verfasser / Verfasserinnen:

Christoph Käferle Max Hoffmann

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Ihr Text hier. Die deutsche Danksagung wird erst verfasst wenn die Englische fertig ist.

Hereby I would like to speak out my thankfulness to our advisor, Dr. Michael Stifter, for his huge amount of patience and understanding with my questions and for giving us the opportunity to dig into one of the most interesting topics in the robotics (humanoid robots). His experience helped us a lot to overcome the big obstacles which occurred during this thesis, in a very helpful way.

In addition, we would like to thank the laboratory "Makers Austria"' for providing us ?Marvin? our humanoid robot.

We would also like to thank our colleges Daniel Honies and Alexander Weller for exchanging usefull information during the implementation. More is yet to come.

Contents

Chapter 1

Introduction

1.1 History of Physiotherapy

One of the origins of physiotherapy which was properly documented was the professional group of Henrik Ling, who was the "Father of Swedish Gymnastics". He also founded the Royal Central Institute of Gymnastics in 1813. The main-focus was set on massages, manipulation and exercises.

In 1887 physiotherapists were given official registration by Sweden's National Board of Health and Welfare. Only a few years later, other countries followed.

Great Britain formed their Chartered Society of Physiotherapy in 1894 and New Zealand caught up in 1913 just one year earlier than the United States. The modern physiotherapy as we know it today, was established in Britain towards the end of the 19th century. It was applied and promoted further during the Polio outbreak in 1916. During the first World War, men & women were recruited to apply physiotherapy to wounded soldiers and help them to return to a normal life. The first professional association within the USA was formed in 1921 and was called the "American Women's Physical Therapeutic Association". This gave birth to what is known today as the American Physical Therapy Association, representing about 76,000 members only in the United States. They define physiotherapy as "Clinical applications in the restoration, maintenance and promotion of optimal physical function".[5]

1.2 History of Robots in the Physiotherapy

The history of robotics in physiotherapy is older than you might think. It started back in the 1980s when the first International Conference on Rehabilitation Robotics took place. Rehabilitation Robotics was introduced first for neurological disorders more than twenty years ago. That was the time, when the need of robots for physiotherapy gained increased attention in the scientific and therapy scene. The first implementation of those robots was not intended for rehabilitation, but rather they were used to help disabled persons who suffered from nervous system disorders. To sum it up, they were not meant to help curing a disorder or injury. They were just designed to comfort the patients.

1.3 Humanoid Robots

In general, a humanoid robot is a structure with its overall appearance based on the human body, allowing him to interact with a person or to act like a person. Mostly, a robot of this kind has a torso with two arms, two legs and a head but this can vary depending on the situation and the needs. Nowadays, human-like looking robots are often seen in movies (Ironman etc.) but those are far away from the reality. There are few humanoid robots commercially available and those which are available, are generally very expensive. The range of possible applications for humaniod robots is endless because they could manage almost every human-powered task if the technology is advanced enough. Humanoid Robots are still in an early development stage with many unknown challenges coming up. It is really time consuming to develop a robot from zero so it should be a goal to build a fundamental robot as a base for further research.

1.4 History of Humanoid Robots

The History of Humanoid Robots goes back to the renaissance where Leonardo da Vinci probably created the very first human-like looking machine. It looked like an armored knight and was known as "Leonardos Robot".
The first so called Robot was a simple robotic soldier who was blowing a trumpet. That machine was created by a German engineer called "Friedrich Kaufman" from Dresden and it was built in the year 1910.
There is an even older mechanical design from 1868. It was a steam powered man who pulled a cart behind himself. From there on the use for humanoid robots was mainly to be an attraction at world fairs and other big events. Unfortunately, they were not used for any scientific purposes whatsoever. From the 1950s onwards, the focus switched to more serious topics like housekeeping and acting in movies. In the 1970s the scientific field focused their attention on robots and created laboratories and facilities to test the feasibility.[4]

1.5 Different Implementations

Today, there are many different approaches and methods to use robotic systems in the sector of physiotherapy. One of the main uses is the creation of an exoskeleton to help a disabled person to walk again after suffering from a stroke or a major injury (In the area of the limb for example). Creating a robotic walker, which supports elderly people or people with walking disorders, is also a widely-used implementation. Those types of robots are more often used than you might think. But the most obvious one is prosthetic. Everybody has seen a prosthesis in their life and that shows how common and accepted they have become nowadays.

1.6 Problems (Danger and Costs)

A big problem with robots, which are working with humans, is that the interaction can be dangerous. There are many parts in a robot which could fail during an interaction, causing an injury to the patient. If the control unit in the servo fails and sends a wrong signal to the motor, the force of the joint pushing/pulling is up to 70kg. The second big barrier for a robot as a physiotherapist is the cost. The total costs of such a product include hardware, software, working hours etc. The hardware part is the most expensive one. Nowadays, a humanoid robot may cost up to several hundred thousand Euros. The fact, that elderly people are not as willing as young people when it comes to working with technology makes it even harder for developers.

1.7 Health Care

Not everyone has the opportunity to get medical assistance when it's required. The best example are third-world countries which are suffering from major medical problems. But the higher developed countries also have shortages in the medical sectors as well. A low income or a bad insurance, and many more reasons are contributing to a sub-standard health service. The patient might have to wait a long time to get a lesson with a physiotherapist. But that's where you shouldn't wait. You should give your health situation priority number one and that's where our thesis comes into place. A humanoid robot like ours could help in many hospitals and institutions for elderly people. Of course, there is way more to think of, but we want to set the first footsteps.

1.8 Where are we right now?

Facilities for elderly people in Austria and probably the whole world except highly developed countries in relation to technology like Japan and South-Korea aren't using enough robots in their health programs. And that's a big issue in such a modern world as we are living in. There are so many physiotherapists in those institutions, but they are just not sufficient to handle all of their clients. This is also caused by the high percentage of old people in our society, and its getting even worse due to our rapidly aging population. As an elderly person, you have the ability to get in touch with a therapist but not as often as you will require it and that's the issue we are facing. Every person should at least have the possibility to get treated in some way.

1.9 Scope

1.9.1 Robot

There are multiple parts we would like to put into the scope of our thesis. The first and most obvious one is the robot itself. It should be kept as simple as possible and low priced as well. But the structural integrity should be high enough to withstand forces from the outside like a human carrying it around. The procedure of repairing should be kept as easy and flexible as possible. We achieved that requirement by printing all the parts with a 3D printer. The weight is also a subject which should be considered. The robot should be portable enough to move it through a hospital or elderly clinic with ease. Therefore the material steel is not an option as it's to heavy and carbon-fiber is to expensive. That's why we have chosen printed plastic. Looking at the technology which is built in, you should consider the price and of course how long it will last before it breaks. There is no need for an expensive controller which saves a lot of money. The reason why we have chosen those servos and the other parts, is explained in section xxxx of this thesis.

1.9.2 Control Software

The Control Software named InMoov Control is designed to be a simple but efficient program which provides basic information such as minimum and maximum values for the multiple joints of the robot. Besides that, the control software should be able to change every value for every servo manually. The ability to calculate a 2D or a 3D movement should be implemented as well. This program should have the main purpose of testing new positions (in the process of creating new gestures and exercises) or re-adjusting values after exchanging a broken part (for example a servo or a plastic piece).

1.9.3 Exercise Software

The core of our product is the Software in combination with the robot which the physiotherapist works with. It's called InMoov Exercise and it is able to load all the exercises and gestures from a library. The library will increase in size and functionality after every session with a therapist. After loading the library, the main information and a picture will be displayed and of course the controls for executing the movements. An additional feature in future could be the addition of a user database. That would have the purpose to display only the features and exercises specific to a particular patient or disorder.

1.9.4 Sessions with Physiotherapists

Meetings with physiotherapists was one of our goals. They help us to better understand the topic and the requirements and adding new exercises is much easier in cooperation with a professional. They are also better at judging whether our project is progressing in the right direction or not. Their information will be used for the further implementations of this thesis.

1.10 Background

Why did we choose this topic for our thesis? Robots are fascinating to write about and are used in many sectors of the modern world. Both of us have great interest in building robots and programming them. We attended the robotics lesson in our school for four years and we have taken part at many global conferences and international robotics competitions. The logical conclusion would be a diploma thesis about that exact topic and adding the possibility of helping people makes it even more attractive for us.

1.11 Outline

In the following sections of this thesis you will read something about the InMoov robot and its parts. In the Implementation-Section we will talk about our different approaches to display and reproduce the exercises as accurately as possible. We will compare two main implementations to demonstrate the differences in the code.
This thesis is divided into x chapters. A brief description for the content of those chapters is as following:

- XX
- XX
- XX

Chapter 2

State of the Art

2.1 Most common controller systems

Most of the current robots are based on a hierarchical distributed control system. In this type of controlling, the system (robot) is organized and divided into local parts each controlled by an individual controller. All the individual controllers are communicating with the main controller to unify their functionality in order to attain the required purpose for the robot.

2.2 Widely known humanoid robots

Humanoid robot development is a rather new topic in the history of robotic research and only a few of them are recognized in the public domain, most of them are only known to electronic enthusiasts. You will be reading a bit about some of the robots which are available to purchase right now. Some of them are very expensive, but they are still important enough to mention. These following three humanoid robots represent the current status of robot research.

2.3 DARwIn-OP (ROBOTIS OP)

DARwIn-OP is a humanlike looking robot which was created at the Virginia Tech University in their Robotics and Mechanism Laboratory. They collaborated with the University of Pennsylvania and a manufacturer in Korea. The main purpose of this robot is for use in the educational sector but you can use him at home as well. At this moment, the second generation of this robot is the most recent. It is slightly more powerful and a bit more affordable. He has about 20° DOF (Degrees of Freedom) and each joint is powered by an individual MX-28 servo. A dual core processor and 4GB DDR3 RAM are powering the whole unit. For communcation, it offers 802.11n Wi-Fi and a Gigabit Ethernet port. The DARwIn-OP 2 (also known as ROBOTIS OP 2) can be purchased for approximately 10000 USD [3].

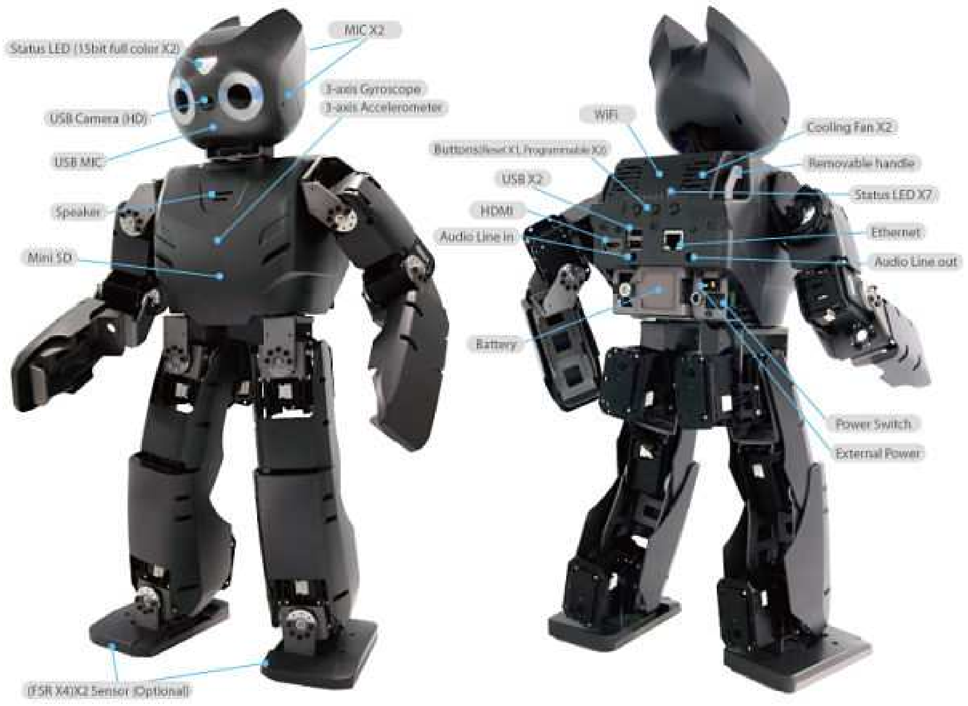

Figure 2.1: A picture of the DARwIn-OP

2.4 NAO Evolution

This is the fifth generation of this specific platform which was developed by the French company Aldebaran Robotics and was released in early 2014. Its height is about 60cm and has a DoF of 25°. The humanoid robot contains numerous sensors such as sonar, tactile and pressure. There are also cameras built in which allows him to perform very well whilst performing complex motions. The fact that the NAO Software is open source, makes it useful for educational purposes or for normal people who wants to add custom functionality to their robot.

The main processing unit is powered by an Intel Atom processor which runs at 1.6GHz running the NAOqi OS. There is even a second controller built in to handle the hardware level functions. By using the cameras and other peripheral devices, the robot can recognize shapes, voices and even people. You can give the robot a command just by saying it out loud which is a pretty nice feature to have. The price tag is currently 7500 USD [8].

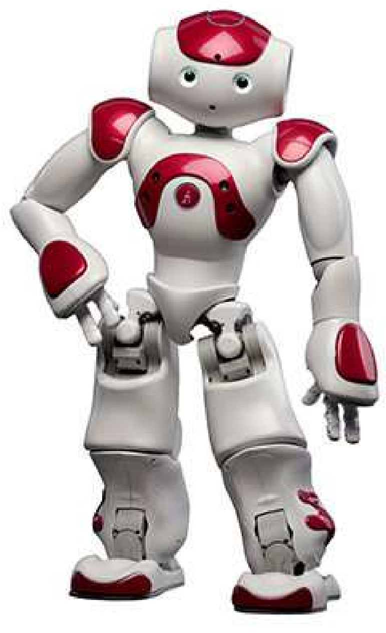

Figure 2.2: A picture of the NAO evolution robot while standing upright

2.5 PR2 Robot System

The PR2 is one of the most developed platforms today. It was created by Willow Garage which is a robotics research company which also developed ROS.

This robot is performing very well in all the important categories. The software is maintained by a large open source community and its specifications and hardware are also very impressive. His height is variable which is possible through his telescopic spine. It can vary between 1.3 and 1.64 meters and his DOF is around 7° He hasn't got the best degrees of freedom but his hands are able to extend to a length of 1 meter. This robot contains many sensors such as laser scanners, Kinect, cameras, pressure among others.

The entire system is powered by two quad-core Intel i7 Xeon processors and 24GB of RAM and several communication interfaces (Bluetooth, Wi-Fi, Ethernet etc.). A 1.3kWh battery provides the power to run the system without independently from extenal power. Due to his incredible electronics, his price tag is unfortunately very high - currently 280.000 USD [9].

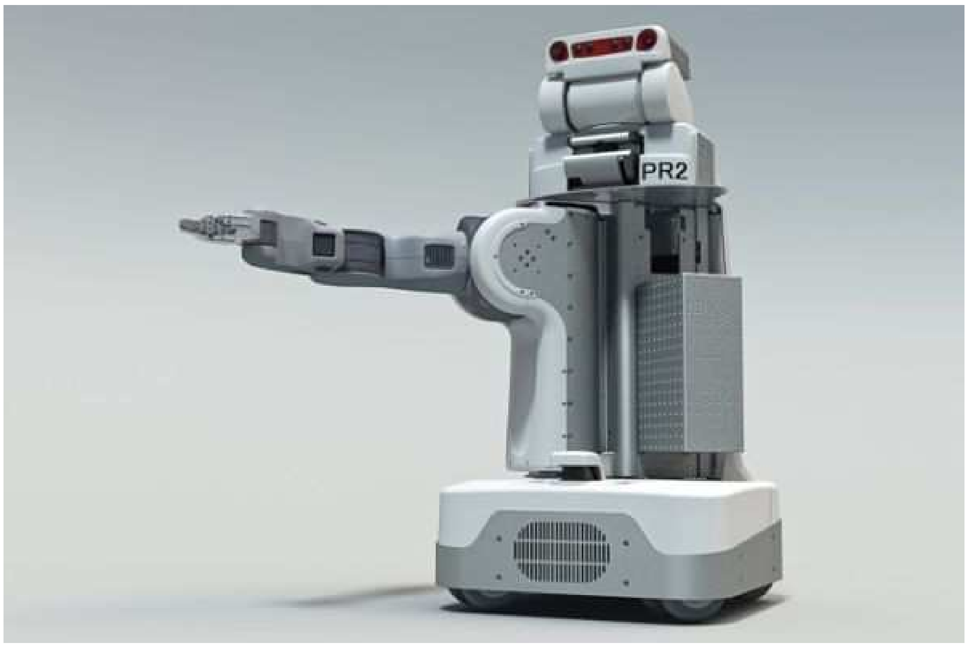

Figure 2.3: The PR2 with its spine at its minimal height and one arm raised

Chapter 3

Marvin

3.1 Summary

The humanoid robot called 'Marvin' is located at the 'Makers Austria' laboratory in Vienna, they built him to provide people the opportunity to work with such a complex machine. The huge benefit of this implementation is the fact that he is as tall as a human being, which provides the possibility to mimic a human performed gesture as accurately as possible.

Weight	30kg
Height	175cm
Degrees of Freedom	9°
Controllers	2x Arduino Uno
Power Supply	max. 20 Amps
Moving Parts	High Grade Servos

3.2 Overview

Each structural part in this robot is printed using a 3D printer which makes it easy to replace a part or to change the design without having any issues with ordering parts online. The fact that the parts are made from plastic also makes the machine as light weight as possible without spending too much money on carbon-fiber or similar hitech materials.

The computing power to calculate the positions comes from a normal PC or Laptop which runs our custom written controlling program and the connection between the computer and the robot is via a standard serial-cable (USB). Two Arduino Uno are receiving the serial information and interpreting them into servo commands to recreate a movement. Splitting the robot

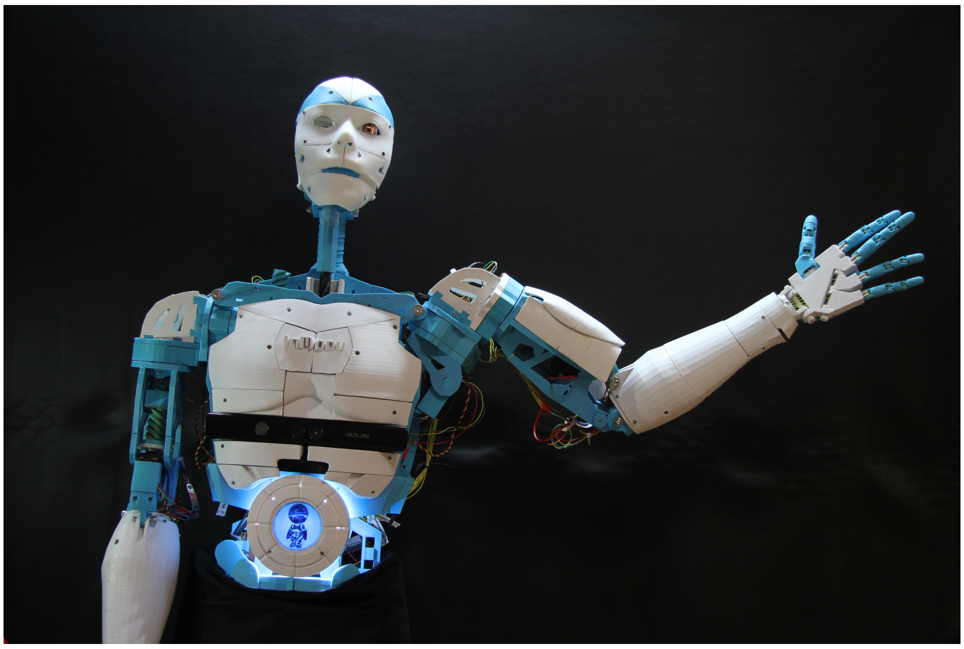

Figure 3.1: The open source humanoid robot InMoov with both arms spreaded out

into physical areas and controlling each with an individual controller is a very common technique. When applying this technique, you need one central controlling unit to connect everything together, which in our case is a Computer. The major reason behind this strategy is reliability. The chance that every controller fails is much lower than the chance when you are using just one.

3.3 Sensors on Marvin

There are only a few sensors which are built into this humanoid robot by default. One of them is built into the chest of the robot and is called Xbox Kinect [7]. The Kinect is a widely known 3D-Scanner distributed by Microsoft for the entertainment system Xbox 360 and Xbox One. It uses lasers and cameras to determine the different depth of a picture to get a 3D-Model via the ToF (time of flight) distance measuring method. This could be used to detect objects and to interact with them with a better accuracy. The second one is a simple camera in the eyes on the head. The camera isn't as advanced as the Kinect but it is good for blob tracking and color recognition.

Those are the sensors which are built in the basic setup of our robot. Of course, some extra sensors would help. For example, some touch sensitive sensors in the palm of the hands to provide the ability to grab objects with the right amount of pressure. Additionally, a few ultrasonic sensors placed around the arms of Marvin would make it much easier to implement the ability of obstacle avoidance.

3.4 Problems

3.4.1 Power Supply

Beside those many benefits, there are also some problems and dangers. We have summarized and described a few of them as short as possible without losing any important information. The best example is the lack of power. It is meant, that one single power supply won't be able to deliver enough energy to make the whole robot move. A more specific scenario would be the one where a user wants to load a gesture which includes almost every servo. The problem is that this would consume way too much energy for our energy supply in a long term and the fact that there are almost 25 servos in the upper body makes it even worse. The best solution for this situation would be adding another power supply, but this would set the price of the whole system higher.

3.4.2 Strength

The second major problem is the enormous strength of the humanoid robot. The main joints are built with industrial grade servos which are providing a maximum strength of 70kg per servo. This means that a wrong move-message to the robot controller could lead to cracking some important structural parts, which isn't that big of a problem (you just print the spare part) but it is much to easy to break it with such a force. There is also the chance of hitting a human-being by accident, which could even be fatal.

3.4.3 Speed

The third problem isn't a major one, it is about the moving speed of the humanoid robot Marvin. For example, the elbow joint. When moving the elbow, the force/torque of the servo is not directly put on the joint. It is put on the spiral which drills its way through the elbow, causing the movement of the arm and this whole process slows it down. The massive weight of the arm is also playing a role in this case. In fact, not all moving parts are slow. The fingers are fast thanks to the strings which are directly transmitting the movement of the fingertip from the servo.

3.4.4 Accuracy

The fourth problem and the last one to be listed, is the lack of accuracy. This isn't such a major problem, considering the fact that our thesis doesn't intend to make gestures where accuracy of a few millimeters is required. For some other applications, this would be a much more serious problem. An example would be another group which is currently working with the very same robot and their intention is to play a board game with this humanoid robot. This accuracy limitation makes it extremely hard for them to fulfill the goal.

Chapter 4

Sessions with Physioterapists

As mentioned in the introduction, we have had conversations and interviews with various physiotherapists with several master degrees. Those meetings were held on the 5th of March 2017. We have split this Chapter into the different sections of the session. Those Sections are described as following:

- Introduction
- The Interviewed Therapists
- Knowledge of robots in the sector of Physiotherapy
- What do they think about our thesis?
- What should be changed/improved in the future?
- Conclusion

4.1 Introduction

Five Physiotherapists attended the meeting on the 5th of March 2017. All of them are currently working in field of Physiotherapy. We met up at the house of one of the therapists and we organised the session as an open discussion. We provided a list below with their names and education.

- Jana Käferle, M.Sc.
- Renate Bilik, M.Sc.
- Birgit Hiebl, BSc
- Stefan Ratheiser, BSc
- Daniela Tieber, BSc

4.2 Knowledge of robots in the sector of Physiotherapy

First of all, every one of the five therapists have heard of some kind of robotics in physiotherapy, but all of them don't really know where robotics starts and where it ends. It was surprising how much they know about some of the new inventions in this sector. The most common answer to that question was the fact that most of the therapists don't have practical experience with any kind of robots or highly developed technologies. Many companies present their developments at conferences to promote them but they don't explain the basic things which leads to misunderstandings and physiotherapists being sceptical which is one reason why robots are not seen that often in their institutes. The budget is also a major factor, when the new investment costs a couple of thousand dollars or even more, but that's one of the things we want to change.

What kind of robots do they know exist for use in physiotherapy? There are a few of them but they know only those, which are highly promoted in their community and conferences which limits the possibility of new partnerships for the competitors. Here is a short overview of those products.

- Lokomat: Treadmill with an exoskeleton to support the weakened body of the patient
- Bioness: Measures the muscular movements from the body and supports him with electric impulses, mounted on the body
- Inno-Walk
- Moto Met
- Hirob: A horse-therapy simulator
- Biotex: A device to measure muscle activities
- Exoskeleton

They had even heard of a development in China, where the goal is similar to our thesis but with a more costly solution using premium materials and top class electronics which is not affordable for everyone.
Virtual Reality (VR) is also an upcoming topic in the eyes of the therapists. The thought behind that is the combination with a physical device to adapt exercises for a better and more efficient treatment. An example would be a VR headset and a treadmill to simulate difficult balance situations.

4.3 What do they think about our thesis?

Renate Bilik, M.Sc. mentioned some of the concerns we have to take care of in the future. For her, the biggest problem is the appearance of the robot because only 10% of the whole treatment process is the right execution of exercises and the rest is a combination of personal contact and the right time table and other stuff, and for that, the appearance of our humanoid robot is contra productive. But this depends on the needs for each individual patient. For some people, a robot without mimic and emotions would be better (autistic people) and some of the kids might have fun with a robot.

Birgit Hiebl said that some of the Therapists might be shy or angry about using such a product, because they see a competitor in their territory. They would be afraid to lose their job if a robot does their own work all the time.

She also stated that the process of adding new gestures or exercises might be too time-consuming for some of the physiotherapists in their daily schedule. The solution for this would be an easy method for creating them. For example, a graphical user interface with a 3D model of the robot where you can drag the joints in the right position.

4.4 What should be changed/improved in the future?

They haven't mentioned any parts which should be immediately changed but there are some things to add in the future to improve the usability and some important and useful features. Therefore, we provided a short summary of those points below.

- Almost every one of the therapists told us that the big robot would fit into hospitals/facilities and other institutions but a human-size robot wouldn't be practical for home-use at all. A small version would make that possible without having a completely new development process. The small robot would be even more affordable and easier to transport. A hospital could give them to the patients who are required do some exercises at home when there is no therapist available. Another way of solving the budget problem would be the implementation of a Desktop-Only version where the robot is just a simulated model.
- After implementing the collision avoidance with the Kinect, the next step in this topic would be the recognition of the patients movement to analyze whether the person is executing the exercise correctly or not. This could be scaled for a bigger group, by setting up a Kinect-Array but it would only work if the people line up parallel to the array.

- To shorten the process of adding tasks/exercises, the addition of a simple puzzle-system would make a big difference. You would simply drag and drop basic movements to create a full task. Many of the therapists (According to the meeting) are not that willing to learn new technologies according to the therapists, but a simple system like this would make it possible for everyone to get the most out of the robot.
- As mentioned above, the look of the humanoid is a bit of a problem. There are multiple attempts which would make the appearance more attractive to the human eye. The first and most obvious thing would be a fake skin made from silicon. The second improvement should be a voice to give him the ability to let people know that theyre not doing the exercise correctly.
- According to Daniela Tieber, measuring the vital data of the patient could help in many ways. For example, to adapt the exercises for the specific person. The client would execute the task as accurately as possible and our system should recognize his limits and adapts it for further sessions. The Vital-Data could also be used for preventing the client from over exerting or fainting.
- Database was a word which was mentioned in the session right at the beginning in context with client specific exercises. A simple database to save the different tasks for every person would make the job easier for physiotherapists. The client would simply log in on the PC which leads into a training which was modified to their current requirements.

4.5 Conclusion

To conclude this Chapter, we summarized the information we gathered during the session and analyzed all the input from the therapists. Basically, the conversation went really well and the feedback was mostly positive. The physiotherapists mentioned a few things to improve and extend which we have described in the text above. They also have shown a lot of interest in our work and said that this development should last longer than just the thesis period. There are so many things to extend the robot and his system with, which make the possibilities endless. This concludes the chapter about the session with the physiotherapists.

Chapter 5

Methodology

5.1 Hardware

5.1.1 Motors

The entire Robot is powered with high grade servos: four in each arm and one smaller servo for every finger. Power and precision were the main factors to determine the motor type to be used. There has to be considered that the motor in charge of lifting the arm vertically needs to lift the weight of all following body parts, therefore it has to be stronger than servos located further down the arm.
There are three types of electrical motors most commonly used in robotics:

- **DC Motor:**
  Through the fact that it has only two inputs (power and ground) it can only be controlled by turning the power on or off. So there is no certain way to determine which position it is in at the moment, except through rough estimations. But in this case the arms have to be moved precisely to avoid collisions.
  Therefore a DC motor can't be used for this project.
- **Stepper Motor:**
  The stepper motor is powered by magnets shifting the motor to an exact position. Therefore, they are more precise than servos but lack the power that is required for lifting one arm of the robot.
- **Servo Motor:**
  Servos contain a DC motor with gears and a potentiometer.The potentiometer measures the current angle at which the servo gear is currently positioned. If the servo motor is used to turn an external transmission the potentiometers data will need further calculations to determine the desired angle value. This type of motor can be controlled by sending the information to which angle it should move. This is possible because the potentiometer gives constant information about the actual

servo position. So the DC motor contained inside of the servo turns until the potentiometer has reached the desired angle.

Considering the power required for lifting one arm and the precision, the servo motor was clearly the best choice.

Servos used within the InMoov Robot:

- BIG SERVO ....
- SMALL SERVO ....

5.1.2 Kinect Sensor V2

In comparison to the old Kinect, the new version provides some major improvements and new features [7].

Figure 5.1: The Kinect Sensor

- **Improved Body Tracking:**
  The enhanced fidelity of the depth camera, combined with improvements in the software, have led to several body tracking developments. The latest sensor tracks as many as six complete skeletons (compared to two with the original sensor), and 25 joints per person (compared to 20 with the original sensor). The tracked positions are more anatomically correct and stable and the range of tracking is broader.
- **Depth Sensing:**
  With higher depth fidelity and a significantly improved noise floor, the sensor gives you improved 3D visualization, improved ability to see smaller objects and all objects more clearly, and improves the stability of body tracking.

  Comparison:

  - The old Kinect has a depth image resolution of 320 x 240 pixels with a FoV of 58.5 x 46.6 degrees resulting in an average of about 5 x 5 pixels per degree.
  - The new Kinect has a depth image resolution of 512 x 424 pixels with a FoV of 70.6 x 60 degrees resulting in an average of about

7 x 7 pixels per degree.

This does not seem as a large improvement, but the depth images of the old and new Kinect cannot be compared that directly. Due to the use of time-of-flight as the core mechanism for depth retrieval each pixel in the 512 x 424 depth image of the new Kinect contains a real measured depth value (z-coordinate) with a much higher precision than the depth image of the Kinect V1. The depth image of the old Kinect is based on the structured light technique. This results in an interpolated depth image that is based on a much lower number of samples than the depth image resolution suggests.

- **Color Camera:**
  The color camera captures full, beautiful 1080p video that can be displayed in the same resolution as the viewing screen, allowing for a broad range of powerful scenarios. In addition to improving video communications and video analytics applications, this provides a stable input on which to build high quality, interactive applications.

  Comparison:

  - The old Kinect has a color image resolution of 640 x 480 pixels with a FoV of 62 x 48.6 degrees resulting in an average of about 10 x 10 pixels per degree.
  - The new Kinect has color image resolution of 1920 x 1080 pixels and a FoV of 84.1 x 53.8 resulting in an average of about 22 x 20 pixels per degree.

  This improves the color image detail with a factor of two in horizontal and vertical direction. This is a welcome improvement for scenarios that use the color image for taking pictures or videos, background removal (green screening), face recognition and more.

- **New active Infrared(IR) capabilities:**
  In addition to allowing the sensor to see in the dark, the new IR capabilities produce a lighting-independent view and you can now use IR and color at the same time.

- **Multi-Array Microphone:**
  Four microphones capture the sound, record audio, as well as find the location of the sound source and the direction of the audio wave.

5.1.3 Arduino

Arduino is a software and hardware company, which set their goal on creating open source controllers for the DIY community. They have started this back in 2005 as a student project in Italy, aiming to provide a low-cost solution to create devices that interact with their environment through sensors and actuators. You can either buy the controller online or build one yourself with the available instructions. Their projects are distributed under the LGPL or the GPL Licenses, which makes it possible for any company to sell their own version of this controller. We are using the Arduino Uno in our project.

Figure 5.2: A picture of the Arduino Uno and its ports

The main focus for this specific controller was set on sensing/controlling objects such as robots. This platform provides a set of digital and analog I/O (input/output) pins which may be connected to various external expansion boards or circuits. The USB-Port (Universal Serial Bus) provides the ability to load programs from your computer. The programming language for Arduinos is a dialect of C and C++ which are the most common programming languages in the robotic sector. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE). [2]

5.1.4 Controller

Why use a micro controller and not a full size PC?
Besides the obvious reason of energy saving in an autonomous system, the entire project is split up in two main parts. One being the robot with two Arduino Unos which are interpreting commands coming from the second part (PC).

Part 1: Robot controller requirements:

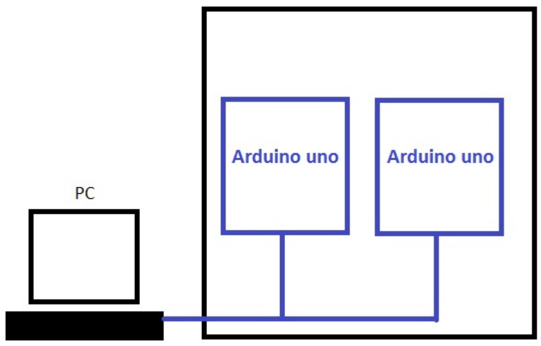

Figure 5.3: The PC is located separately from the Arduinos and connected via USB)

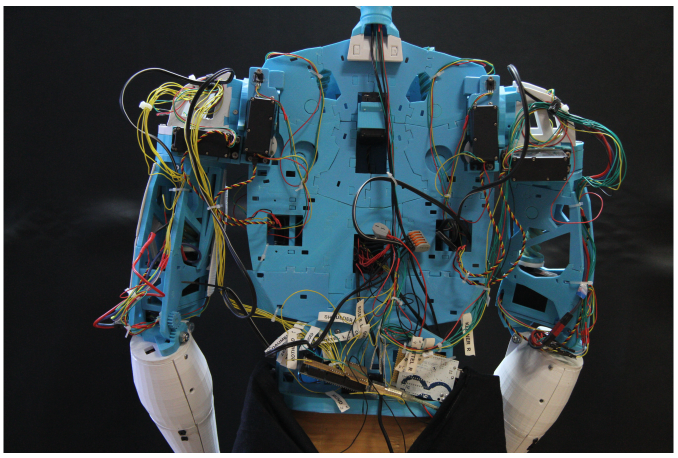

Figure 5.4: The back of the InMoov robot

- Controller has to have a small form factor to fit onto the robot. Consequently the weight has to comply with the strength of all the materials used in the robot.
- Low Hardware requirements (CPU usage?)
- Needs more than 20 ports to connect servo motors to. One digital pin (data pin) per servo and one shared ground and voltage supply.
- Has to support well tested environment for programming with electrical components.
- Needs serial port to communicate with the PC

Two different platforms were considered: Arduino Uno and Raspberry PI 3 Model B.

The Arduino platform was chosen for this project because it doesnt require any operating system to be installed on it and it offers a wider variety of ports to control all servos. Arduino delivers a complete package including the micro controller with pre-installed operating system and an IDE for windows and mac users. It doesnt have as much processing power as its direct competitor the Raspberry PI but due to the fact that in our projekt the controller just needs to interpret data from the serial port there is no need for a powerful processor.

Part 2: PC requirements:

- Needs to be able to process data quickly for fast response of the robot. This is needed to assure there is no damage caused to the robot itself or surroundings due to a slow response of the robot.

The PC has just one requirement, being able to run the Arduino IDE which is necessary to push code onto the Arduino.

5.1.5 Robot Parts

When building a humanoid robot, the choice of the material affects the robots' stability and the weight determines how strong the electrical motors have to be.
The plans which we used to build the robot are from an open source project initiated by a French designer. This project started in January 2012 with the designer Gael Langevin publishing the 3D plans for a prosthetic hand. Throughout the years more and more parts were added to the project until it evolved to the first ever complete open source project to print a humanoid robot [12].

5.2 Software

5.2.1 Algorithmic

The main goal is to make it as simple as possible to control the robot. There are two main parts:

- **2D calculations**
  To make controlling the robot as easy as possible we planned to add a 2D coordinate system. In which a certain point has to be chosen and using those coordinates and knowing the dimensions of the robot, there are algorithms to calculate those values.

- **3D calculations**
  The problem you have when working with a humanoid robot is you have to establish some kind of normed system containing all joints and

their current locations inside a 3 dimensional coordinate system. Every joint has a current angle measured in degrees from its resting position. But when working with a replica of a human arm there are multiple joints on each arm and if one joints angle is changed, every joint which is connected to this limb gets moved to a new position in the coordinate system.

It should also be taken into account that the joints in a human arm are ball joints but at the current state of robot technology a human ball joint has to be separated in to multiple joints. Most implementations use three joints to mimic a human joint like this. Which leaves one joint for each dimension of movement (x-axis, y-axis and z-axis).

5.2.2 Kinect SDK 2.0

"The Kinect for Windows software development kit (SDK) 2.0 enables you to create commercial or Windows Store apps and experiences that support gesture and voice recognition by using C++, C, Visual Basic, or any other .NET language or Windows Store projection. The integrated developer toolkit includes sample applications with access to full source code, Kinect Studio, and resources to simplify and speed up application development."[6]

5.2.3 Collision Avoidance

5.2.4 Serialization

Definition

"Serialization is a process for converting a data structure or object into a format that can be transmitted through a wire, or stored somewhere for later use."[10]

Methods

When data is stored or sent it's mostly serialized to assure a standardized format. There are a variety of different formats to serialize data but there are some which are most often used. Each with pros and cons depending on the situation requirements.
Some of the most widely used serialization formats:

- Json (JavaScript Object Notation)
- XML (Extensible Markup Language)
- YAML (Yet Another Markup Language)

When deciding which format to use many factors come into play. For instance the overhead which is created or in other words additional data which has to be stored to determine characteristics like the data type after deserializing.
For the communication between both programs used in the InMoov project there were two formats taken into account. Firstly the Json format and the Google protocol buffers.
Why is Protobuf better than the Json format in our case? Protobuf provides a specification language to define a schema which ensures consistent data on the contrary the Json format doesn't provide any unified data type or structure control.

Listing 5.1: Json example

```
{"Person": {
          "id": "1",
          "name": "max",
          "email": "test@sample.com"
     }}
```

Listing 5.2: protobuf example

```
message Person {
  required int32 id = 1;
  required string name = 2;
  optional string email = 3;
}
```

As seen in the code example 5.1, Json doesn't provide a definition for any data types. Thus any data serialized from one client can be of any data type. Which requires many user input checks on the second client which receives the Json message. Because there can always be some data loss when transmitting data and in the worst case the data becomes corrupted and there has to be checking if the object was serialized properly.

With the Google protocol buffers (example 5.2) the data type is clearly defined therefore the data has to be in a correct format when being serialized as well as when converted back to a protobuf object [1].

5.2.5 Protobuf serialization

Protocol buffers were originally used internally at Google to deal with a server request/response protocol. Now the protocol has been released to the public and supports several programming languages for instance c++, c, go, java and python. It's based on a data scheme which defines the structure of the message to be sent in a file with the file extensions ".proto" as seen in the code snippet 5.2.

After creating the ".proto" file the protocol buffers provide a data type matching the previous definition. Every field is supplied with a setter and getter.

Listing 5.3: protobuf getter and setter

```
Person person;
//setter
person.set_name("Max");
//getter
person.name();
```

5.3 Coordinate System

5.3.1 Overview

There are two different types of coordinate systems required for the task of 3D-calculations when working with a robotic model. The global coordinate system is our standardized reference frame to compare all other systems. It describes the location of all joints and the end effectors in three dimensional space. Local coordinate systems are the second type used in kinematics. There is a local reference frame per joint. Coordinate systems are also referred to as a reference frame.[11]

5.3.2 Global Coordinate System

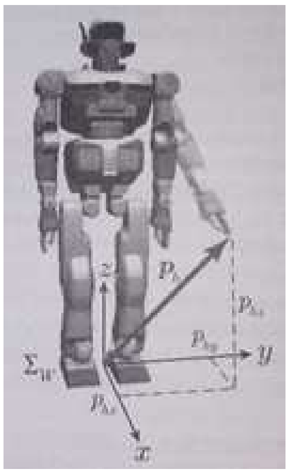

Figure 5.5: $\sum_W$ is the global coordinate system

When adding every limb and joint to the global system further calculations become simplified especially regarding collision detection or determining if a point is within reach of the arm. Because these actions can be reduced to simple value comparisons without any transformations being required. Positions within the global system are referred to as absolute positions.

5.3.3 Local Coordinate System

For each joint there is a local coordinate system as seen in figure 5.6 which has a vector displaying its position in relation to the origin of $\sum_W$ as well as a rotation depending on the state of the joint [11]. All local coordinate systems have dependencies to their predecessors. This means that if one joint is moved, all following joints are moved to a new absolute point in our global

reference frame.
The following is an example how a local coordinate system changes in relation to the global reference frame.

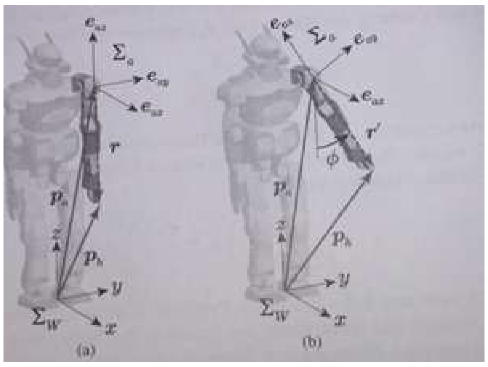

Figure 5.6: $\sum_W$ is the global coordinate system. $\sum_a$ describes the local reference frame for the elbow joint

The vector Pa points to the origin of $\sum_a$ from the view of the global coordinate system. $\sum_a$ is initialized parallel to $\sum_W$ but after the rotation of the shoulder joint the $\sum_W$ system is rotated be the angle Due to $e_a x$ being the axis of ration, $e_a x$ doesnt have to be rotated.

5.3.4 Transformation Matrix

This matrix is used to transform a point from one coordinate system to another. When working with kinematics this matrix is used to transform points in the local coordinate system to the global reference frame. Homogeneous Transformation Matrix is a part of the forward kinematics which are used to locate the end effector in the global coordinate system when all angles of the joints are known.[11]
This matrix is needed to convert the coordinates of a point located in the local system $\sum_1$ to its absolute location in $\sum_0$.

The homogeneous transformation matrix (Ta) consists of two sub matrices which are the Rotation Matrix (Ra) and the displacement vector (Pa). These have to be calculated before building the finished transformation matrix.

$$\mathrm{T}_a = \begin{bmatrix} & \mathrm{R}_a & & \mathrm{P}_a \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.1)$$

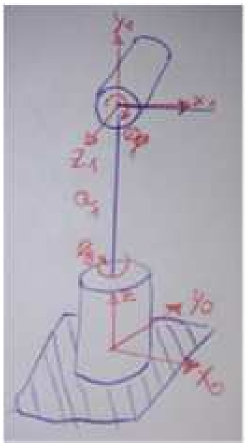

Figure 5.7: Example to calculate a transformation matrix to translate between $\sum_0$ and $\sum_1$

The rotation matrix 5.3.4 defines how $\sum_0$ has to be rotated to have the same orientation as $\sum_1$. In this case of the x-axis already points in the same direction so it has to be rotated around the x-axis by 90 degrees to match $\sum_1$.

$$\mathrm{R}_a = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad (5.2)$$

Every column represents one axis of the second coordinate system. To rotate the base system ($\sum_0$) we have to enter the number 1 if the axis of the base system is orientated in the same direction as the second coordinate system. If the axis is pointing in the opposite direction a -1 has to be entered. For instance, the z-axis of $\sum_1$ is equal to the inverted y-axis of the base system. The -1 indicates that y has to be inverted. To complete the rotation matrix, the matrix on the left has to be adjusted to the angle of rotation which is theta. This is done by multiplying both parts of the rotation matrix. When multiplying these matrices every row of the left hand matrix gets multiplied by each column of the right hand matrix. Which leaves us with the finished rotation matrix for $\sum_0$ to $\sum_1$.

$$\mathrm{R}_a = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ \sin(\theta) & 0 & -\cos(\theta) \\ 0 & 1 & 0 \end{bmatrix} \quad (5.3)$$

The last step for a finished transformation matrix is to calculate the displacement vector which points to the origin of $\sum_1$.

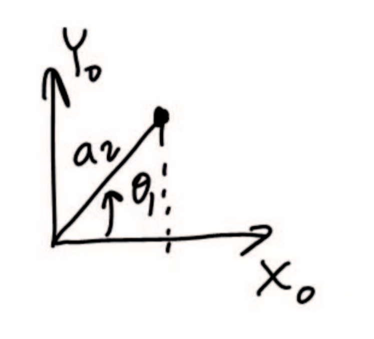

Figure 5.8: View from above onto the reference frame $\sum_0$ from Figure 5.7

View from above $\sum_0$ To get the displacement vector we have to calculate three variables (x,y,z) by by which the local coordinate system $\sum_1$ is displaced in relation to the $\sum_0$. For the x and y value we view the base frame $\sum_0$ from above and calculate the x value using the law of cosine, same procedure with the y value the only difference is here we have to use the sinus law.

$$\begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & \mathrm{a}_2{*}\cos(\theta) \\ \sin(\theta) & 0 & -\cos(\theta) & \mathrm{a}_2{*}\cos(\theta) \\ 0 & 1 & 0 & a \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad (5.4)$$

5.3.5 Transformation Matrix Chain Rule

The local coordinate systems in a body part, for instance the hand can be seen as a chain of coordinate systems. As a result of that the position and orientation of the n-th joint in the global reference frame can be calculated by multiplying the transformation matrices from $\sum_0$ to $\sum_n$ [11].

5.3.6 Degrees of Freedom

Degrees in freedom are defined as number of coordinates which are required to determine the position of a body in space. Therefore, an object in a three

dimensional space has 3 degrees of freedom. But in the case of a human hand it can also be tilted in relation to the x, y and z axis. These three rotation movements around x, y and z are called roll-, pitch- and yaw rotation. Which leaves us with 6 degrees of freedom in the case of a human hand.
Each of the three rotations around the axes have their own rotation matrix which is used in the forward as well as the inverse kinematic.
x-axis = roll y-axis = pith z-axis = yaw

$$\mathrm{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{5.5}$$

$$\mathrm{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{5.6}$$

$$\mathrm{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.7}$$

5.4 Kinematics

5.4.1 Denavit-Hartenberg convention

This is the most commonly used model to calculate 3D Kinematics when working with robots.

This convention sets three rules to make this procedure possible [11]:

- The Z-axis has to align with the rotation axis of the joint.
- The X-axis has to lay on the common normal of Zn and Zn-1 .
- The Y-axis has to apply to the right hand rule.

5.4.2 Inverse Kinematics

Inverse Kinematics is used to calculate all the angles of a model when the coordinates of the end effector are known. This technique is the exact opposite to forwards kinematics where all the angles are known.
This procedure exceeds the scope of this thesis so it is not implemented. But the following text is going to give a brief overview of the methods available. There are two methods most commonly used in robotics. One being the analytical method which is the direct way to calculate all angles and the second on the numerical method.

Analytical Solution

This approach is the geometrical way to calculate all the angles of, for instance the arm of a robot. When at the beginning of this procedure the only parameters known are the coordinates of the end effector as well as all measurements regarding the length of body parts. Based on this information a triangle can be drawn. For instance the vertices could be the end effector, the location of the shoulder joint and the elbow joint. Due to the fact that all of the measurements except the angles are known, the law of sines can be applied to calculate the missing angles.
BIILD
This procedure has to be continued to get the roll and pitch angles. The complete solution for the analytical method has been described in [13]. Most humanoid robots are implementing the human ball joints by using three rotational joints. As a result of this implementation the three rotational joint axis aren't meeting in one point, which over complicates the analytical method. In this case the numerical solution should be used.

Numerical Solution

Chapter 6

Implementation

The entire project has been split up in two components as seen in **??**, the PC which is processing all the user input and doing the calculations regarding collision avoidance and inverse kinematics. On the other hand, the two Arduinos which are receiving orders how the servos should be moved. Both parts are connected together via a USB cable.

6.1 PC Software

Author: Christoph Both of our PC Projects are implemented in C# and were written with the programming environment Visual Studio. For the UI, we have used WPF. The major benefit of WPF is the fact, that it is already built into Visual Studio which guarantees us a seamless work experience and we haven't noticed any problems which were caused by this tool. To improve the user interface of our programs programs we have chosen a 3rd party developed extension because the standard WPF look seems a bit generic and old school. During the implementation phase, we paid special attention to the readability of our code, so anyone could modify it to their specific requirements. We kept the names of the variables logical and included comments where they are needed.

6.2 PC Exercise

We tried to make this program as simple and easy to use as possible. The reason for this is due to the user groups we expect to use the software. The main users will be the therapists/clients and not technicians who are familiar with more complex solutions/user-interfaces. The standard view supplies the main functions such as loading and executing exercises.

When opening the program, the login window will be displayed. Afterwards you will be redirected to the exercise tab where you can choose the preferred gestures/exercises. You can choose from a predefined list of all gestures and after choosing one, the main information and a picture will be displayed for a more detailed description.

SCREENSHOTS

6.3 PC Control

This project was implemented to give the users a tool which is suitable for some quick testing and movements. For example, when you are testing a new gesture or exercise. The user interface is split into several tabs. All those tabs are deactivated when starting up a new movement, except the first one which is the connection-tab. In there you must put in the serial connections which are used to connect to the robot and after you have successfully connected, you have access to all those features. Tab 2-6 is for the several components on the robot (left and right arm/hand and the head). In those tabs, you will find control inputs for every single servo so you can change them exactly as you require.

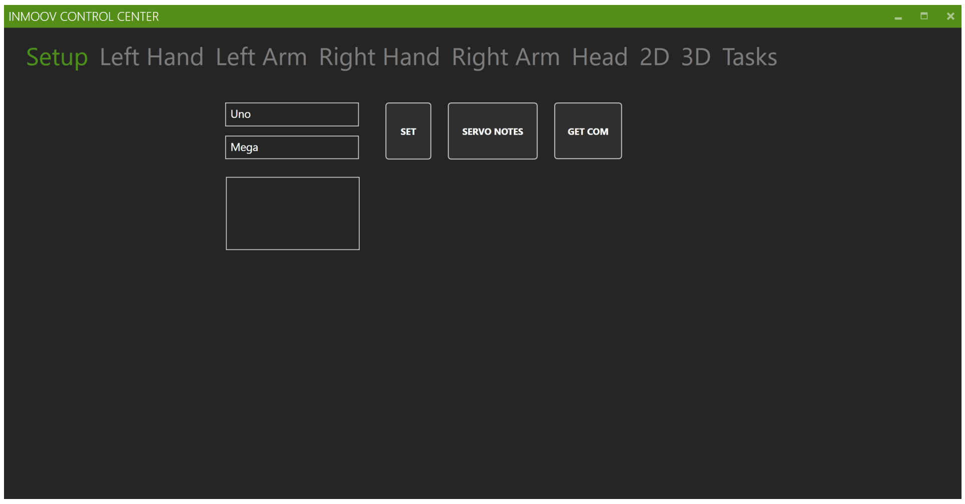

Figure 6.1: The Login Screen from the Control Program

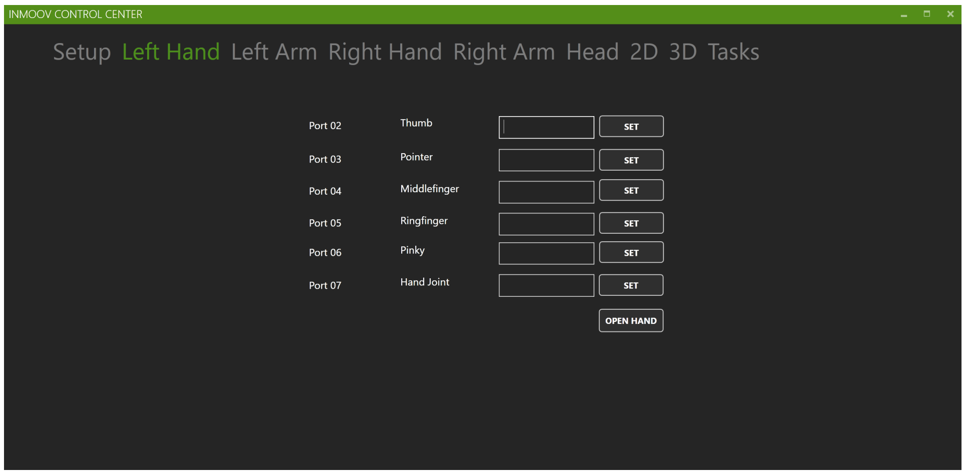

Figure 6.2: An Example for a Controlling Tab (In this case for the left hand)

The interesting tabs are number 7,8 and 9. Tab 7 is providing the ability to calculate a 2-Dimensional movement. For that you must put in the x and y coordinates. The next tab does the exact same thing except that the coordinate system is now 3-Dimensional. The last one is kept simple but still useful. A drop-down list is provided to list all gestures which are saved in the project. To conclude this section, the controlling program can provide all the same features as the exercise software and also much more. But not all the users will want to work with such a powerful tool and that's where the second program comes into place.

6.4 Saving the Gestures and Exercises

We had to come up with a way to save the gestures and the exercises. One of the most common ways to save objects and simple data structures is XML (Extensible Markup Language). The syntax of XML is really easy and efficient to use. That's why we have chosen this technique for our implementation.

With this markup language, you have to define an object structure according to your needs and requirements. In our case, it was a list of gesture/exercise objects with three attributes. The first attribute is the name, and that's what's displayed in the programs when searching for a gesture (for example in the drop down list of the gesture tab in the control program). The second attribute is the most important one. It holds all of the servo positioning information which is required for the execution. Depending on the fact if the movement is affecting one or both sides of the robot, the syntax of this attribute differs. If it's just one side, it starts with a letter according to the side which is moved, so if the gesture is for the left side, the first character is "L". After that, the positioning information begins. Every servo needs 5 characters and only those are written down, which are required to be moved. The complete position-attribute looks like something similar to: "R08135090601105010100" (This specific example would move the right arm in the right position for shaking hands). If the gesture is on both sides, it would look like : "L07090080900909010095110000 R07090080900909010095110000". We also provided a short example of our XML-Code below.

Listing 6.1: XML Example

```
<Gesture>
   <Name>Arm Reset</Name>
   <Positions>L07090080900909010095110000 R07090080900909010095110000</
   Positions>
   <Info>Both Arms will be reset to the standard position!</Info>
 </Gesture>
```

6.5 Collision Avoidance

TODO

6.6 Serial Protocol

To delegate as much of the required processing power to the PC program as possible, the delegation of left and right hand movements are split up between the two arduino controllers. The PC decides if the command should go to the left and/or right arm and based on this decision the message is sent over only the required serial connections. This allows both Arduino programs to be almost identical.
There are just two pieces of information which have to be transmitted from the PC to either one of the Arduinos to complete a movement, the port number which the servo is connected to and the angle it has to move to. As a consequence, the actual servo position has to be saved on the PC side. All the ports from top to bottom have been assigned to each servo going from the shoulder down to the wrist on each side to make it intuitive which port moves which body part.

Joint Name	Port Number
Omoplate	11
Shoulder	10
Rotate Arm	9
Biceps	8
Wrist	7
Pinkie	6
Ring Finger	5
Middle Finger	4
Index Finger	3
Thumb	2

6.7 Arduino Program

This part of the program has been kept very simple. The only task the Arduino has to perform is to separate the information packages from each other. Every pair of port number and angle has exactly 5 digits, that? how the program can pick out the data. After reading every block all servos are moved simultaneously.

6.7.1 First Implementation

Simple string serialization was the first attempt.
To send two pieces of information (port number, angle to move to) the simplest way is just to combine the digits. So a message to move the biceps to an angle of 60 degrees would be "08060" The first two digits being the port number and the last three being the degrees to move to. Advantages of this method are that there are no libraries required and it doesn't get affected by different programming languages. In this case a c program (PC) and a c/c++ program (Arduino) are communicating over string serialization. The downside of this type of messaging is that everything depends on the constant length of the messages. For instance if the project is extended and there is a need for more than 99 sensor ports. If the message template gets rewritten the entire program on both ends has to be adjusted accordingly to the new message length.

Listing 6.2: first serial implementation

```
if(inputLength>5 && inputLength%5 == 0){
  int indexCounter = 0;
  for (int i=0; i < inputLength; i = i+5)}
    servoList[indexCounter]= input.substring(i,i+2).toInt();
    angleList[indexCounter]= input.substring(i+2,i+5).toInt();
    indexCounter++;
  }
}
```

As seen in the code snippet 6.2 the code becomes rather complicated, therefore it's another source of unnecessary errors caused by string processing and casting into other data types.

6.7.2 Second Implementation

The second attempt to solve the communication between the PC and Arduinos was to rely on a trusted protocol. Google protobuf (protocol buffers) are used to serialize structured data defined in a .proto file more to *protocol buffers* in [1].

Arduino protocol buffers

The previous implementation of reading one or multiple servo commands was well above 100 lines of code. All of that has been reduced to about 20 lines of code including the following code snippet for deserializing the protobuf class.

Listing 6.3: Google protocol buffers reading from serial stream

```
 proto::ServoCom servoMessage; //instance of protobuf object
//deserialize protobuf object if possible
if(!servoMessage.ParseFromIstream(strm)) {
      Serial.println("Failed to read sensor");
 }
 Serial.println(servoMessage.portNum()+" "+servoMessage.angle();
```

6.8 The User Database

We have implemented a user database, so we can choose which gestures and exercises are suitable for the different clients. After thinking about an easy way to implement it, we made the decision to use xml, considering the fact that there won't be any user sensitive data saved. The conclusion for that decision is the fact, that we are not using a password to login. You simply use the username and nothing else which saves us the need to implement encryption.

Listing 6.4: Example of a Data Entry in the UserDB

```
 <User>
    <Name>Max Mustermann</Name>
    <Birth>01.01.1990</Birth>
    <Height>1.65m</Height>
    <Adress>2500 Baden, Austria</Adress>
  </User>
```

Chapter 7

Experiments

7.1 Accuracy of the humanoids arm

The fact, that a physiotherapy-exercise must be executed as accurately as possible, makes it important for our thesis to have an experiment on that exact issue so we are able to present some valuable data for further research. After considering some ways to implement that experiment and discussing about them, the easiest/ best way was to execute a gesture a few times in a row while measuring the coordinate of the fingertip. This gives us the ability to create a table for all those measurements and to analyze the data for some interesting results in accuracy.

7.1.1 Set-Up

To measure those differences in movement, we had to come up with an experimental setup. The best way is to keep it as simple as possible. Placing the robot in a room with enough space to move was the first required step. After that, we needed a static point in our global three-dimensional coordinate system. To achieve that, we placed a pole vertically in front of the robot in the position we have previously chosen. On the top of the pole, a colored ball was mounted as a reference for the end of the movement of the humanoids arm.

7.1.2 Execution

First of all, we implemented a gesture in our program which moves the arm of the robot to the coordinate where the ball is mounted. Afterwards, we simply repeated that procedure a few times to get more data to analyze. We measured the distance to the fingertip of the robot to the ball on the top of the pole after executing the movement. We wrote down the x/y/z coordinates of the fingertip while keeping the ball fixed as a reference.

7.1.3 Results

Table with description text and some information.

	x	y	z
1	0,5cm	0,7cm	0,9cm
2	0,5cm	0,7cm	0,9cm
3	0,5cm	0,7cm	0,9cm
4	0,5cm	0,7cm	0,9cm
5	0,5cm	0,7cm	0,9cm
6	0,5cm	0,7cm	0,9cm
7	0,5cm	0,7cm	0,9cm
8	0,5cm	0,7cm	0,9cm
9	0,5cm	0,7cm	0,9cm
10	0,5cm	0,7cm	0,9cm

7.1.4 Causes

7.1.5 Conclusion

7.2 Reliability of the collision-avoidance

Author: Hoffmann Max
This experiment builds up on the previous experiment about the accuracy of the robots arm.(reference auf experiment) Which showed that the accuracy of the servo motors used in the inmoov robot aren't accurate enough to rely on calculations for collision avoidance. The other possibility to prevent collisons with the other arm was to use visual recognition. Using this method the algorithm has to be fast enough to detect future collisions in real time and the accuracy of the robots movements doesn't influence this approach. Which makes this the only feasible solution to prevent collisions and damage to the robot itself.

7.2.1 Set-Up

The experiment consists of two parts. Firstly the robot repeats a motion where the two arms come very close to each other (about 5cm apart) during those repetitive movements the kinect is setup in different locations. Then we evaluate the success rate of detection. Because different angles through which the camera and sensors used by the Kinect see the hands and arms of the robot greatly affect the detection rate of which the body parts are recognized.

7.2.2 Execution

7.2.3 Results

7.2.4 Conclusion

7.3 computing time protobuf and binary

Every command sent to the robot has to be serialized and de-serialized. The standard serializing method used in c++ takes up a lot more processing power than for instance the Google protocol buffers. A custom test situation is needed to quantify how much processing time is saved when switching to protobuf.

7.3.1 Execution

7.3.2 Results

	time
1	11μs
2	5μs
3	4μs
4	12μs
5	9μs
6	6μs
7	9μs
8	4μs
9	10μs
10	12μs

	time
1	11μs
2	5μs
3	4μs
4	12μs
5	9μs
6	6μs
7	9μs
8	4μs
9	10μs
10	12μs

	time
1	17μs
2	20μs
3	6μs
4	6μs
5	19μs
6	9μs
7	6μs
8	20μs
9	10μs
10	12μs

	time
1	11μs
2	5μs
3	4μs
4	12μs
5	9μs
6	6μs
7	9μs
— 8	4μs
9	10μs
10	12μs

Bibliography

[1] https://www.developers.google.com/protocol-buffers/docs/overview, Accessed: 09.03.2017 19:00.

[2] *Arduino uno.* https://store.arduino.cc/product/A000066, Accessed: 11.2.2017 16:00.

[3] *Darwin-op (robotis op).* http://support.robotis.com/en/product/darwin-op.htm, Accessed: 20.12.2016 15:00.

[4] *History of humanoid robots.* http://wonderfulengineering.com/history-of-humanoid-robots-in-pictures-1868-1970/, Accessed: 12.10.2016 14:00.

[5] *History of physiotherapy.* http://www.chiropractorswarwick.co.uk/index.php/a-history-of-neuromusculoskeletal-healthcare/a-history-of-physiotherapy-physical-therapy/, Accessed: 12.10.2016 12:00.

[6] *Kinect sdk 2.0.* https://developer.microsoft.com/en-us/windows/kinect/tools, Accessed: 13.03.2017 14:00.

[7] *Kinect sensor v2.* https://developer.microsoft.com/en-us/windows/kinect/hardware, Accessed: 13.03.2017 15:00.

[8] *Nao evolution.* https://www.ald.softbankrobotics.com/en/press/press-releases/unveiling-of-nao-evolution-a-stronger-robot-and-a-more-comprehensive-operating, Accessed: 20.12.2016 15:00.

[9] *Pr2 robot system.* http://www.willowgarage.com/pages/pr2/overview, Accessed: 20.12.2016 15:00.

[10] Cline, M.: *Serialization and unserialization.* http://www.parashift.com/c++-faq-lite/serialization.html#faq-36.3, Accessed: 09.03.2017 20:00.

[11] Kajita, S.: *Humanoide Roboter*. 2007, ISBN 978-3-89838-079-9.

[12] Langevin, G.: *Inmoov humanoid robot.* http://inmoov.fr, Accessed: 16.03.2017 15:00.

[13] Paul, R.: *Robot Manipulator*. 1984.

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —

Breite = 100 mm
Höhe = 50 mm

— Diese Seite nach dem Druck entfernen! —