University of Southern Queensland
Faculty of Engineering and Surverying

The Development of a Computer Based System for the Rehabilitation of Arm Movements of Stroke Patients

A dissertation submitted by

Nathan Prasser

in fulfillment of the requirements of

Courses ENG4111 and ENG4112 Research Project

towards the degree of

Bachelor of Engineering (Mechatronic)

Submitted: October, 2008

Volume 1

Abstract

Strokes are a very commonly occuring medical event. In Australia alone around 53,000 people suffer from a stroke each year. A stroke occurs when the normal blood flow to the brain is interrupted causing sections of the brain located around the affected areas to become starved of oxygen and nutrients. The damage caused to the brain during a stroke can leave the affected person with different types of mental and physical disabilities. It is currently known that the physical disbalilities that can result after a stroke can be overcome by the use of physical therapy allowing a stroke victim to continue a normal life.

The aim of this project was to design and build a computer based system that could assess and exercise the upper extremities of stroke patients. A literature review revealed that the most successful outcome that has been seen with experimental physical rehabilitation is to combine constraint induced therapy with robot-aided neuro-rehabilitation.

The system that was designed over the course of this project operated within a two dimensional field, therfore only two degrees of freedom were required. The movements of this two link system were measured and logged with the use of two precision potentiometers, one located in each rotational joint. This allows all movements made by the patient to be later recalled to assess progress the patient has made. Improvements would be determined by comparing the path the patient has taken compared to the optimum path between targets.

Overall the system had very high precision and was able to obtain the required work area for sufficient exercise to be completed. The GUI developed in coordination with the stroke technology is simple to use and bug free. The GUI was designed to be very adaptable to the requirements for any user by adding as many functions as possible while still remaining simple.

The future work required on this system would include further researching the tracking technology and possibly incorporating a more accurate potentiometer. The system currently offers passive exercise, the addition of other technolgies (e.g. DC motors) has still yet to be introduced to allow active movement. Suggestions of possible technologies and designs to incorporate this function are discussed throughout this dissertation. The GUI software should be further developed to allow automatic improvement calculations to be performed, and displayed in a user friendly manner e.g. percentage of improvement.

Certification

Nathan Robert Prasser

Student Number: 0050025790

____
Signature

____
Date

Acknowledgments

I would like to thank the following people:

- Dr Selvan Pather for his support, guidance and dedication given during the process of this dissertation.

- My family for all the support they have given me over the past year.

Table of Contents

Volume 1

Chapter 4 Prototype Development & GUI

Chapter 5 Device Evaluation & Analysis of Results

List of Figures

List of Code

Abbreviations

TIA:	Transient Ischaemic Stroke
DVT:	Deep Vein Thrombosis
T-WREX:	Therapy-Wilmington Robotic Exoskeleton
MIT:	Massachusetts Institute of Technology
F-M:	Upper limb subsection of the Fugl-Meyer
MP:	Motor power for shoulder and elbow
MS1:	Motor status score for shoulder and elbow
MS2:	Motor status score for wrist and fingers.
AutoCITE:	Automated CI Therapy Extension
DOF:	Degrees Of Freedom
CEO:	Corporate Executive Officer
ADC:	Analogue to Digital Converter
GUI:	Graphical User Interface
VB.NET:	Visual Basic.net
CAD:	Computer Aided Drafting
PC:	Personal Computer
DOS:	Disk Operating System
OS:	Operating System

Chapter 1 Introduction

1.1 Introduction

In the past century we have seen a rapid advancement in technologies able to assist in making our lives easier. These technologies are often taken for granted now as our culture has become more adapted to using them every single day. This could be as simple as the invention of telecommunications or transportation by motorized vehicles. There is one major field were technology is only just starting to have an impact as computer technology is becoming cheaper, faster and more accepted. This is in the field of medical robotics, robotics that are used to assist surgeons operate, diagnose and rehabilitate patients.

The field of interest relating to this dissertation is the rehabilitation of a patient who has suffered paralyses to the upper extremities caused by a stroke. This following chapter will outline and describe the purpose of this study. Further discussion of the problem that is being faced and the objective of the study will be described in detail.

1.2 Outline of the study

A stroke is a commonly occurring medical phenomenon. Each year over 53,000 people suffer from a stroke in Australia alone. It is the second largest killer and one of the leading causes of disability [1]. Four out of ten people who survive a stroke are left with a disability. The resulting physical disabilities are currently treated with physical therapy. This project is to research and develop technology that would be able to be introduced into the rehabilitation system to allow stroke patients to recover from paralysis of the upper extremities. This rehabilitation process will concentrate on improving three functions.

- Perception
- Coordination
- Muscle strength

1.3 Stroke Technology

Several technologies are currently being developed around the world either through private companies or universities to assist in the rehabilitation of patients who suffer from upper extremity limb impairments. The technology developed for rehabilitation of the upper extremities all follow some basic rules governed by the current rehabilitation techniques which will be discussed in Chapter 2. For the technology to be accepted by society there are high expectations placed on the development of a new technology to assist in providing improvements that are not easily accomplished by current techniques. This is where the use of computer based technology becomes a necessity to outperform human dependant rehabilitation.

There several benefits with using a computer based technology for rehabilitation. These benefits are as follows:

- Computers are able to store large amounts of information perfectly for decades.
- Computers do not become bored or loose focus while performing repetitive tasks.
- Once a computer based technology has been introduced into a system they are extremely cost effective.
- A computer based technology will not suffer from muscle fatigue.
- It is able to offer many different variations to be compatible with any user.
- It is able to use the technology available to convert the repetitive exercises associated with rehabilitation into a video game.

There are many different designs currently being researched and development. Although their designs differ greatly they all have one purpose in mind, this is to create the most successful system possible to outperform other technologies available. Some systems currently only operate within a two dimensional field, this is best described as a bench top device. By operating within two dimensions the system is significantly simplified making manufacturing cheaper. Other systems work within a complete three dimensional environment. This allows the user to exercise more muscle groups.

The most beneficial outcome of involving robotics in the rehabilitation system is that as many robots as required would be able to be purchased allowing people who have recently suffered from a stroke to be immediately introduced into a program to start rehabilitation. Without robotics the number of patients that would be able to be treated would be limited by the number of physical rehabilitation professionals available.

1.4 The need for this study

There are only a handful of technologies which have been developed to assist in the rehabilitation process. Dr Erin Lalor, CEO of the national stroke foundation said “stroke survivors tell us that not enough is being done to help their recovery when they return home from hospital” [1]. Technology could play a vital role in overcoming the difficultly of leaving the safety of a hospital to return home. By developing an instrument that is simple to use and the technology is affordable this could be used within one’s own personal environment, mental development would be able to be involved in their rehabilitation process being the patient is able to practice their daily exercises away from the hospital.

Another important development in using technology for rehabilitation is being able to convert what seems to be a difficult and repetitive task into a game. There are few people that enjoy completing the repetitive exercises that are required for rehabilitation. By incorporating video games into the development of stroke technology would help make the procedure of physical rehabilitation a simpler task for both the patient and the person assisting them.

Current rehabilitation involving no computer technology to monitor progress relies heavily on the accuracy of the physical therapist for progress reports and determining the path of the exercises in which to follow for maximum success. By integrating a computer system into the rehabilitation process to record all movements, progress would be able to be monitored closely and accurately, this will result in the highest possible success of rehabilitation and eliminate human error from the process.

1.5 The Problem

In current society technology has been developed for almost all tasks to make our lives easier. Examples of this could be the introduction of cars for easy transport, or telecommunications introduced to makes local and worldwide communications simple. But there is one area were technology has had little involvement to date. This is the introduction into a physical rehabilitation program for disabled patients such as stroke patients.

This research was undertaken to develop a system that can be introduced into a successful physical rehabilitation system. As seen from current rehabilitation techniques there are many factors that must be included into my design to create a successful system. As the system will be required to be introduced into a physical rehabilitation program successfully and with a smooth transition as there will be several initial design factors which need to be accounted for. These factors include:

- The system must be lightweight, if the system does become heavy then an additional system must be introduced to simulate a lightweight system.
- The system must be able to be moved freely through all anticipated locations of use. A system with points of free movement and points of restricted movement would be unacceptable.
- The system should be affordable but still have reliable components which are easily manufactured and easily repaired or replaced. An expensive device would be difficult to introduce into a rehabilitation program with a smooth transition.
- The device should be able to be used within most environments. This means that the system would only require a standard power supply, and a simple computer.

1.6 Research objectives

The research objectives are as follows:

- Research the cause and underlying physical effects experienced after a stroke has occurred.
- Research current physiotherapy techniques that are being used to combat loss of limb control.
- Research current technologies which are currently being used to assist in rehabilitation.
- Design and build a model which will be able to move in any point within two planes of movement.
- Do initial programming of arm to calculate arm location.
- Modify arm to be able to provide resistance and assistance.
- Program arm to provide resistance and assistance.
- Design and build a bench for the arm to operate on with LED's embedded to create objectives for user to follow.

1.7 Methodology

Below is the method planned to allow the most successful rehabilitation technology to be developed.

- Investigate current rehabilitation techniques

  The first step to achieving my goal is to investigate techniques which are currently being used in stroke rehabilitation. By knowing this information I will be able to develop a system that can reproduce a similar routine. The main goal of developing a technology to assist in the rehabilitation process is to outperform current rehabilitation techniques that are available today. Therefore we must introduce into our system techniques or technology that will better current human rehabilitation.

- Brainstorming

  This leads directly to the process of brainstorming ideas. Incorporating computer technology allows for the perfect preservation of data which plays a crucial role for the creation and modification of rehabilitation routines to achieve the highest possible outcome. Brainstorming has allowed the creation of new ideas combined with technologies that could produce an effective rehabilitation device.

- Rough designs

  We now can start drawing rough ideas on paper to see what kind of device could fulfill the above requirements. When multiple sketches have been completed the designs which appear to be realistic and able to handle all the above requirements can be modeled in a computer environment such as Pro/Engineer Wildfire. This allows the designer to view their ideas more clearly, allowing obvious faults with the system to be removed or redesigned before further design or construction is commenced.

- Choose design

  Once all designs have been modeled I will then be able to choose a model which seems to best fit the requirements. This design can be further modified to make the system more efficient, and to be cheaply/easily modified if required. Once the chosen design is completed, designs can be taken away to be built.

- Software development

  During the build process software can begin development to handle the system. For a computer system to be able to handle the physical requirements I have chosen to use a development board. In this case we are using the LPC-MT-2138. This board will be discussed further in chapter 4.

The LPC-MT-2138 will be used mainly for the ADC capabilities and also some simple math routines. The GUI will be developed using the VB.NET software. Once the hardware has been constructed, the hardware and software can be combined.

- Analysis of results

  When the system is fully functioning the results can be analyzed to determine how successful the overall design has worked.

1.8 Dissertation Overview

Chapter 2: Strokes & Stroke Therapy

This chapter reviews physical therapy techniques currently being used. It covers several types of physical therapy from human involvement therapy to robot aided therapy and its resulting fatigue on the human body.

Chapter 3: Conceptual Designs

This chapter introduces the initial concept designs that I have developed. This includes general advantages and disadvantages of each system.

Chapter 4: Prototype Development & GUI

This chapter shows detailed information relating to the chosen design. It also talks further about the software and related hardware used to accomplish my goal.

Chapter 5: Device Evaluation & Analysis of Results

This chapter will review the developed system and describe the successes and failures of the current design.

Chapter 6: Conclusion

This chapter concludes all other chapters giving a final result and possible changes to the current design.

1.9 Conclusion

Strokes are a commonly occurring medical phenomenon not only within Australia but worldwide. There is strong evidence that physical disabilities caused by strokes can be overcome by a physical exercise program. This physical exercise aims to rewire different sections of the brain to control limbs which are currently suffering from paralysis. Limb paralysis is a common occurrence resulting after a stroke and is due to the death of brain tissue within the section of the brain related to motor control functions.

At present the majority of all physical exercise programs rely on human interaction for successful rehabilitation. There are several systems being developed at present to assist in physical therapy by robotic automation. This investigates the current systems and steps through the procedure which I have undertaken to develop a system which I hope will be successful in assisting the evolution of medical rehabilitation technology.

From this chapter we have established some fundamental requirements of the system. These are, the device must provide a suitable exercise technique, provide assistance and resistance to movements made, log and analyze patient movements and the system will only operate within a two dimensional field. A suitable method has also been established to allow the task to be completed. The following section will discuss technical details of strokes and the current techniques used for rehabilitation.

Chapter 2
Strokes & Stroke Therapy

2.1 Introduction

This chapter discusses technical details relating to strokes and will review many relevant publications. Reviewing these publications will emphasize the purpose behind the research which is currently being undertaken throughout this dissertation.

The publications which are reviewed throughout this section concern many different topics. Such topics are, constraint-induced movement therapy, the pain and fatigue relating to the intensity of constraint-induced movement therapy, robot-aided neuro-rehabilitation and the effects of undergoing short term robotic therapy.

2.2 Strokes and Rehabilitation

The following section covers the technical details of the types of strokes that can occur as well the physical influences that increase the possibility of a stroke happening. This section also discusses the process of quickly introducing a stroke patient into a rehabilitation program, some technologies that are currently found in the rehabilitation process and the overall goals that the rehabilitation process is trying to achieve.

2.2.1 Strokes

A stroke occurs when the normal blood flow to the brain is interrupted. This interruption results in the brain being unable to receive oxygen and nutrients required for normal operation. The section of brain which is associated with the stroke will die. There are two types of strokes, an Ischaemic stroke and a Haemorrhagic stroke.

An Ischaemic stroke occurs when the blood supply to the brain is blocked. This blockage can be caused by a blood clot or a fatty piece of material becoming wedged within a blood vessel. Blood clots form in the arteries, these clots can break free and travel through the entire body. Another phenomenon to consider is transient Ischaemic strokes or TIA. TIA strokes are similar to Ischaemic strokes but differ by time. A TIA stroke (also referred to as a mini-stroke) is the interruption of blood flow to the brain for a short period of time.

A Haemorrhagic stroke occurs when a blood vessel or an aneurysm ruptures. An aneurysm is “a disorder of the heart or arteries in which the wall bulges outwards at an area of weakness” [2].

Aneurysm's can be caused by an injury, disease or an abnormality present at birth. In most cases this rupture causes bleeding inside the brain and in rarer situations the bleeding can occur on the surface of the brain. These two situations can be seen in figure 2.1.

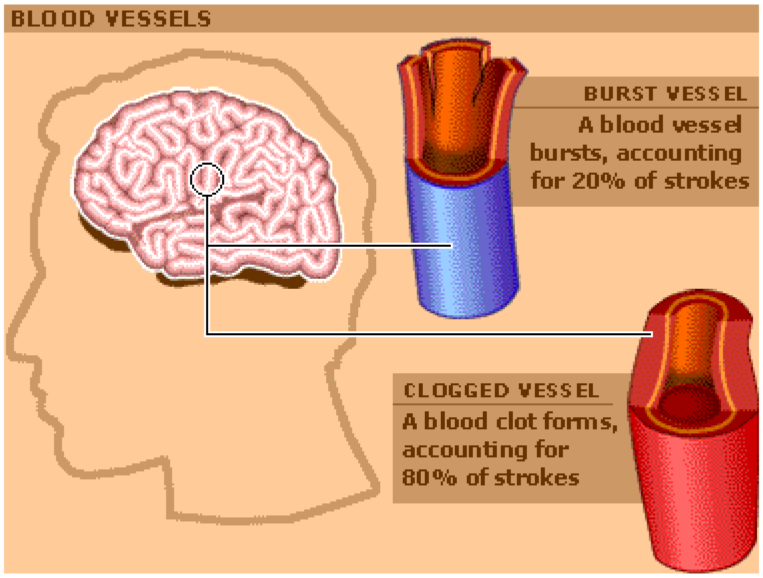

BLOOD VESSELS
BURST VESSEL
A blood vessel bursts, accounting for 20% of strokes
CLOGGED VESSEL
A blood clot forms, accounting for 80% of strokes

Figure 2.1: Ischaemic and Haemorrhagic strokes [3]

Strokes can affect people in any age group, but are more common in older people. Studies indicate that 40% of people under the age of 65 years old are at risk of having a stroke, of these the people below the age of 45 years old account for only between five to ten percent [4]. Those older than 65 years of age are up to 25 times more likely to suffer a stroke [5]. Figure 2.2 shows the deaths resulting from strokes relative to age in the United States during 1998.

Older people are more prone to suffer from a stroke due to their body aging. As you age your arteries narrow, harden and fatty deposits can accumulate on the walls of arteries which carry blood to and within the brain. These fatty deposits are called Atheroma. Narrowing of arteries drastically increases the chances of a clot occurring, resulting in an Ischaemic or TIA stroke. Some unhealthy habits which have been linked to aid in the degradation of the arterial system are an unhealthy diet, smoking, excessive alcohol and some illicit drugs which weaken the walls of the arteries.

Some resulting complications that occur from a stroke can be either physical, mental or a combination of both. The complications are determined by the area of brain affected and the severity of the stroke. Some physical complications affect speech often resulting in speech impediments. A more concerning complication is paralysis to some or all of a person's body. Paralysis is one of the most obvious signs that someone has had a stroke. This is visible due to the lack of muscle control and is most obviously seen in the face and extremities such as arms and legs.

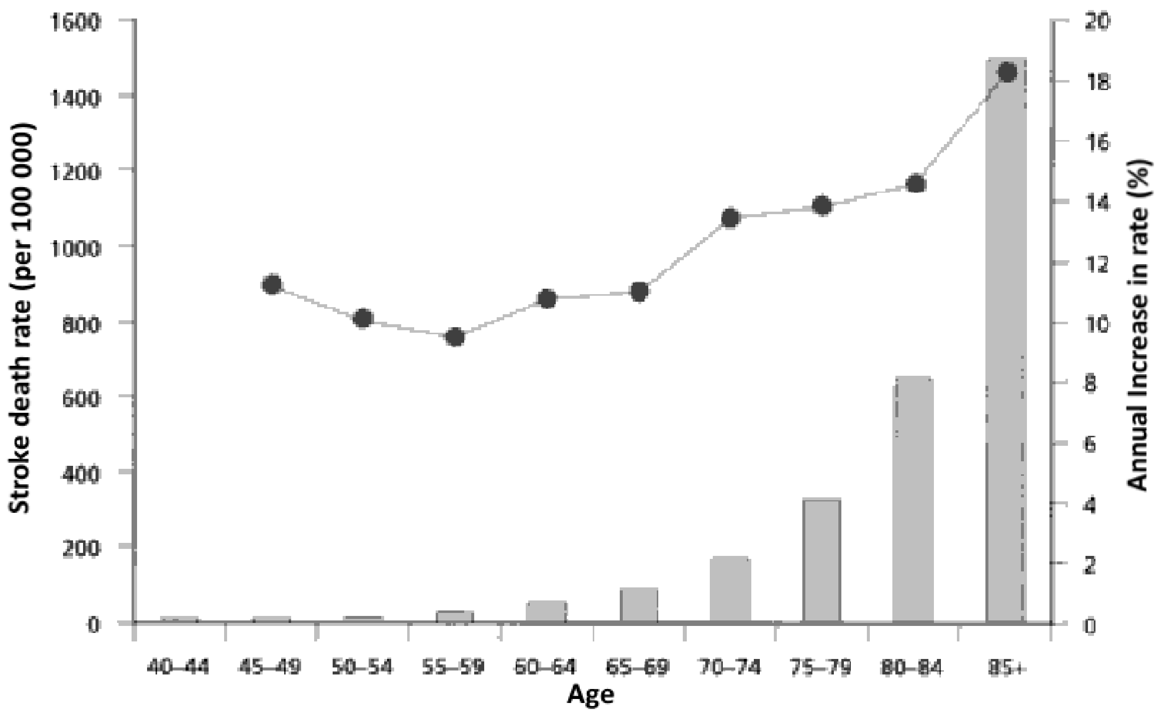


Figure 2.2: Histogram of death rate per 100 000 [6]

The immobility caused by a stroke has direct implications on a person's physical and mental wellbeing. Depression is a commonly occurring mental result following a stroke. Other physical and mental related health complications are occurrences such as pneumonia, deep vein thrombosis (DVT), memory loss, seizures, limb pain and problems understanding language [7].

2.2.2 Stroke Rehabilitation

The direct physical effects resulting from a stroke currently have rehabilitation programs developed to achieve the most successful results possible. These rehabilitation programs involve many differing professionals each specializing in a specific area. These professionals include physicians, rehabilitation nurses, physical therapists, occupational and recreational therapists, speech-language pathologists and vocational therapists.

Physical therapy is a process which is undertaken quickly after a stroke has occurred. This is usually within 24 to 48 hours after a patient has stabilized [8]. Physical therapy is a process to teach the brain motor control which was lost during the stroke. The direct implication of a stroke occurring within brain is tissue material dying, the section of tissue death is related to the location of the clot/rupture. The tissue that has died will always remain so and will not regenerate. Physical therapy is a process of teaching a different section of the brain that has not been affected by the stroke to control the now paralyzed limb. There are many techniques which are currently used that have been shown to be successful with the rehabilitation of muscle control.

Many experts recommend the most successful technique required in any rehabilitation program is well focused repetitive tasks. These are often tasks completed everyday such as regaining the ability to dress oneself, bathing and walking. At first most stroke victims will have no control of specific limbs relating to the section of brain affected by the stroke and its severity. This lack of control requires a physical therapist to assist in regaining movements in these affected limbs. This technique is referred to as active movements. All movements during physical therapy are within a controlled environment and are practiced within a certain range-of-motion.

As the patient progresses and regains some limb control the physical therapist will let the patient do more while they assist less. This process will continue until the patient requires no assistance, at this stage exercises are referred to as passive movements. To regain full motor control the exercises must be appropriately designed to work the affected muscles. During rehabilitation complex movements must be practiced, an example of complex movement is the ability to coordinate leg movement and body positioning to walk. As the patient progresses and exercises are repeated motor control will return.

2.2.3 Current Rehabilitation Technologies

Currently there are only a handful of robotic technologies being used in the medical industry for physical rehabilitation. One such technology which was developed at the University of California-Irvine was a robotic arm called the Therapy-Wilmington Robotic Exoskeleton (T-WREX) [9]. T-WREX was designed purely for the rehabilitation of stroke victims with indirect therapist supervision, and was specifically designed for upper extremity training.

Another technology designed for rehabilitation of stroke victims was a product from MedGadget [10]. MedGadget developed a joystick with a force feedback system, similar force feedback systems are common within many gaming joysticks developed today. The introduction of force feedback into the joystick allowed MedGadget to develop software that would allow patients to physically feel virtual terrain. The user would be required to move the joystick in a specific direction, if the movements made by the patient were incorrect the force feedback joystick would be able to correct it. Repeatedly using this system would allow a patient to learn precise movements and regain motor control.

2.2.4 Rehabilitation Goals

For a patient to be physically able to complete tasks required in everyday living there are several areas that rehabilitation must focus on. Everyday tasks refer to the jobs an able bodied person takes for granted. These everyday jobs could include brushing your teeth or dressing yourself. Therefore to be able to complete these tasks a person must acquire three main capabilities. These are as follows:

- Perception

  Perception describes the process whereby sensory stimulation is translated into organized experience [11]. Perception is very important for a human allowing us to react accurately to a situation correctly. An example would be trying to do something as simple as making a coffee. Being unable to perceive the distance between the kettle and mug correctly could easily become a dangerous situation resulting in the patient being burnt by spilt water. Other difficulties associated with perception difficulties are [12]:

    - Noticing things that are on one side of the body.
    - Recognising familiar faces.
    - Recognising colours.
    - Understanding what you see.
    - Getting around without getting lost of bumping into things.

- Coordination

  Coordination is the process of being able to control the correct muscle groups to move a limb in the desired motions. An example of coordination is being able to use both hands together to button a shirt. This is a very simple example but in actual fact requires a high degree of coordination being it requires small and accurate movements.

- Strength

  Strength is the ability for a person to exert a force through the use of muscles. Strength is the most important factor associated with rehabilitation, without the strength to move the effected limb the two previous factors cannot be trained. The ability to move the effected limb even slightly will allow for further development of strength, perception and coordination. Patients unable to move their effected limbs at all will be provided assistance until some motor control returns. An example that requires strength is being able to grab an item off the shelf at a grocery store.

2.3 The Effectiveness of Unilateral and Symmetrical Bilateral Task Training

This section covers two types of human dependant rehabilitation. One type is the conventional upper extremity arm training while the other uses some experimental techniques in the same rehabilitation program. The experimental program uses standard rehabilitation technique for majority of the program but offers patient new techniques involving unilateral and symmetrical bilateral task training.

The study [13] that was undertaken involved 41 patients selected from a group of 176 candidates according to their physical abilities. Patients were rejected if they experienced any of the following:

- No upper extremity motor return.
- Cognitive disorders.
- No upper extremity disabilities.
- Severe aphasia.
- Cerebellar stroke.
- Bilateral stroke.

The 41 patients that were eligible for the program were divided at random into two groups. The controlled group practicing the conventional therapy contained 21 patients while the experimental group contained 20 patients. Passive and active movements were practiced initially in both groups. Examples of exercises practiced in the controlled group were:

- Putting blocks or cones in a pile.
- Unscrewing a light bulb.
- Shuffling playing cards.
- Putting a pillow in a pillowcase.
- Tearing sheets of paper.

The experimental group performed these same tasks for majority of the rehabilitation time. For 45 minutes four times a week over the five week program the experimental group would be involved in symmetrical bilateral and unilateral exercises which involved exercises requiring the coordination of both arms. Examples of these exercises are

- Wringing a garment
- Rolling a cylinder

- Making a coffee

Both groups were given an equal amount of time over the entire program to practice the exercises that were set out for them to use. Their progress was evaluated using several tests. Motor control was evaluated by Fugl-Meyer, grip strength was evaluated with the Martin vigorimeter, gross manual dexterity was measured by using the box and block test, fine manual dexterity was measured using a pegboard test and motor coordination was measured using the finger to nose test.

After the five week period the results showed that both groups had progressed. Analysis of the results did not show any statistical difference between the two groups. From this research we have gained a better understanding of current rehabilitation techniques being used and experimental methods being incorporated into these programs in an attempt to improve the patient’s rehabilitation status.

2.4 Constraint-induced movement therapy

This section will cover one current physiotherapy technique currently being used in the rehabilitation of stroke patients. This technique is referred to as constraint-induced movement therapy and is classified as intense physiotherapy compared to other techniques [14].

Constraint-induced movement therapy is a technique where non-affected limbs are restricted forcing the patient to use their equivalent paralyzed limb for most of their daily activities. This means that if the patient was suffering paralysis on one side of their body in the upper extremities, (or more refined their arm) then the patient’s fully functioning arm would be restrained.

This might seem like a dangerous practice but the restraint used during this process is designed so the arm can easily be removed from the restraint in the case of an emergency. The restraint is positioned around the hip of the patient so the limb would sit naturally in a comfortable position. There are some exceptions to the rules, and patients are allowed to use their functioning limb for some activities. In total 90% of daily activities should be completed using the disabled limb. Using their functioning limb for the other 10% of daily activities avoids aching and stiffness from occurring to their able arm.

Past studies have shown that most improvements will occur during the first 11 weeks following a stoke [15]. After a period of six months to a year little to no recovery will occur [16]. The article mainly under review in this section involved 26 patients who have suffered from a range of medical injures causing brain tissue death resulting in paralysis of limbs. The 26 patient’s involved ranged from 22 years of age to 67 years of ages with an average age being 52.5 years.

The time from which the stroke occurred to beginning of the program ranged from four months to 22 years. This aspect of the research would test previous standards for rehabilitation timelines.

During the rehabilitation process the patients would undergo three separate checkups. There was an initial checkup that would set a benchmark of the patient's current abilities. The patient would then return at two weeks and six months for a checkup and analysis of progress.

An individualized program was developed for each patient according to their disabilities and their own personal goals they wished to achieve in the program. Each patient would complete a set of tasks during a six hour period, this would be repeated everyday of the week. If a patient was unable to complete the tasks within the set period of time between 09:00-15:30 (which included a half hour break) they were advised to finish the tasks at home. After six months the information was examined and tabulated.

The results from the study [17] show that improvements have occurred in every field. Most significant improvements were achieved during the first two weeks of the program. The information obtained during this research contradicts the traditional view of rehabilitation for patients suffering disabilities similar to those who participated in this program.

2.5 AutoCITE

There are several experimental technologies that have been developed that lead in the field of biomechatronic rehabilitation for stroke patients. One of these technologies is the AutoCITE system [18]. The AutoCITE system consists of eight separate exercise devices which are as follows:

- 1 – Reaching

    Reaching consists of a target being displayed on a touch sensitive computer monitor. The patient must reach out and touch the target. Upon touching the target the object will change to a new location.

- 2 – Peg board

    This exercise involves the patient placing different sized pegs into a board. This helps the patient develop coordination skills.

- 3 – Supination/Pronation

    Supination and pronation requires the patient to rotate a handle in one direction until it stops moving in that direction, then they reverse the direction of movements until it stops once again. Each time that the patient successfully completes the rotation to its full extent its ability to rotate further will increase. The patient must achieve as many repetitions of this exercise within a set time period.

- 4 – Threading

  Threading with the AutoCITE system is the same as threading a needle except this is performed on a larger scale.

- 5 – Tracing

  Tracing makes use of the computer touch screen. A shape will appear on the screen and the patient must trace this pattern with their finger.

- 6 – Object-flipping

  A patient will be timed as to how many times they can flip an object.

- 7 – Finger tapping

  Finger tapping consists of the patient's fingers being place over touch sensitive buttons. They will then have to tap one of their fingers in coordination with an instruction they are given from the computer.

- 8 – Arc-and-rings

  The patient will hold a ring in their hand, this ring has two DOF. One DOF is a rotation about a fixed point, the other being a slide allowing the radius of arc to be changed. The patient must move the ring within a two dimensional environment to coordinate the ring through an imaginary point.

These eight exercises can be seen in figure 2.3.

The entire AutoCITE setup consists of a computer monitor, a desk with four rotational work platforms (each platform containing two exercises each) and a chair fixed along a track that is attached to the desk. This setup can be seen in figure 2.4.

(Discussion continued on page 18)

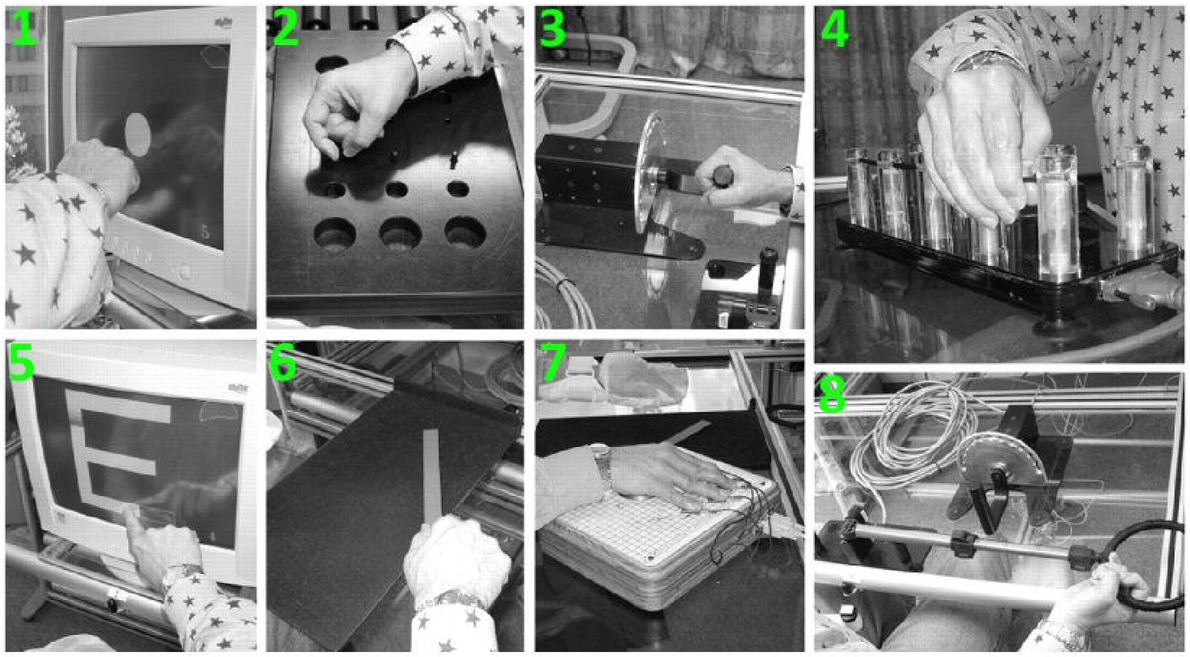


Figure 2.3: AutoCITE Exercises [18]

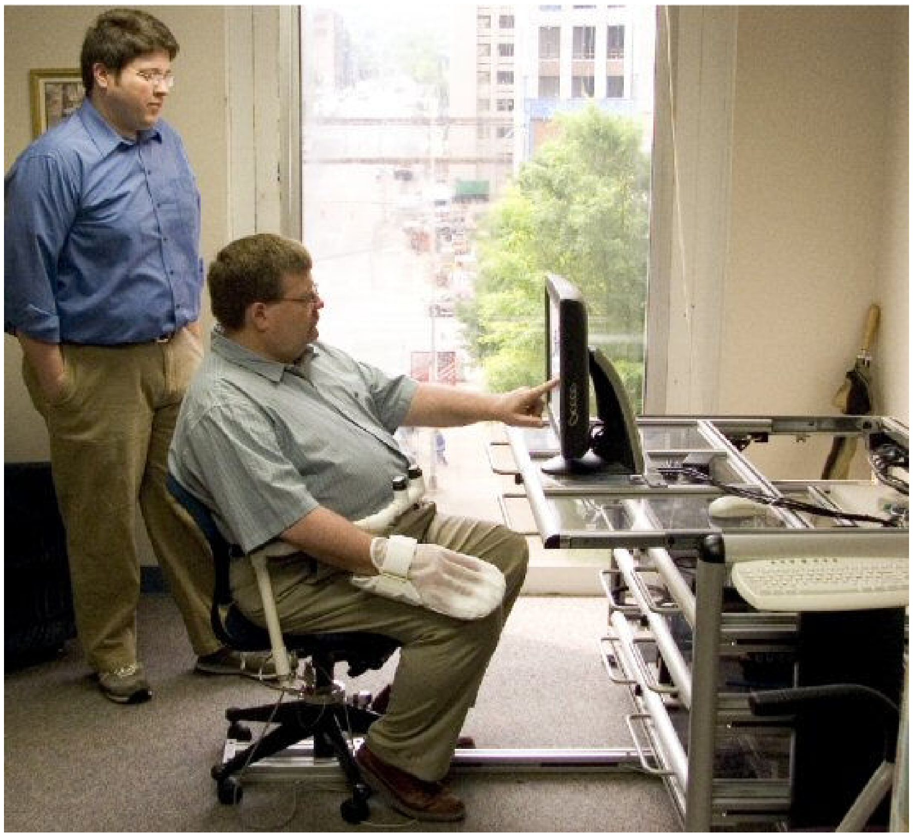


Figure 2.4: AutoCITE Arrangement [19]

Twenty-seven patients underwent a trial [19] of the technology, each patient suffering from a mild to mild/moderate stroke. They were assigned into one of three groups. One group spent 100% of their rehabilitation time with a physical therapist, another group spent 50% of their time with a physical therapist and the third group only spending 25% of the time with a physical therapist. This was to test if the patients who spent less time with the physical therapist would regain as much motor control as compared to the group with larger contact hours.

AutoCITE stroke therapy is classified as CI therapy therefore the non-effected limb was restrained for 90% of the day's activities. This was seen in figure 2.4 above. Patients underwent three hours of AutoCITE training a day for a two week period. The exercises consisted of ten 30 second trials where the objective of the exercise was repeated as many times as possible. Once this was complete the patient would then be able to begin a new exercise.

Results of the experimental trial showed no significant difference between the three groups. Therefore we are able to obtain the same positive results with less contact with a physical therapist. This in turn makes the rehabilitation process more affordable.

2.6 Pain and Fatigue Associated With Movement Therapy

Physical rehabilitation after a stroke is extremely important for a patient to regain control of paralyzed limbs. The most beneficial results found from the previous study were achieved during the start of the program, this was a short but intensive period of physical exercise. The significance of this study [20] was to determine if the physical exercises performed during the rehabilitation period take a toll on the patient's physical wellbeing and if these exercises are causing adverse symptoms.

A study was undertaken to test if there was any correlation between the common symptoms of a stroke patient and the rehabilitation program. The symptoms which were being closely monitored were pain and fatigue induced by the repetitive tasks. Since the research concerned rehabilitation induced injuries, a strict policy for patient involvement was required. Patients could only enter the program if they suffer from minimal pain. The patients must also have minimal elbow, hand and wrist activity in a range-of-motion in extension.

A total of 32 people were involved in the program, 22 participants were men with an average age of 64 years old. The other 10 participants were women with an average age of 57 years. Over a two weeks period the patients exercised for 10 of those days with an average of 4.5 hours dedicated to exercises each day.

Immediately following therapy there was no significant increase of pain with any patient. Any significant pain that was reported during the therapy was found not to be associated with the exercises. All reported pain associated with the exercises during the day was classified as minimal. It was reported that as the patient's motor control increased so did the pain. As the day progressed fatigue levels increased slightly. Over the course of the program it was shown that there was a relationship between fatigue and pain. From this study it can be seen that there are adverse symptoms associated with the rehabilitation program. These symptoms are minimal and could be expected even with an able bodied person performing the same physical exercise in the rehabilitation program.

2.7 Robot-Aided Neuro-Rehabilitation

Robot-aided neuro-rehabilitation for upper extremities of stroke patients is the main research interest of this project. This section will be based around discussion of current research results obtained from physical tests performed. A study [21] was conducted from 1995 to 1999 that compared the results of physiotherapy obtained from two groups. One of these groups would be involved in a standard physiotherapy group which would involve physiotherapy only from a physiotherapist. The other group would receive physical therapy from not only the physiotherapist but on top of this they would perform an extra 4-5 hours a week using a robotic arm designed to rehabilitate physical impairments of the upper extremities.

This study involved 76 patients all suffering paralysis caused by a stroke. Both groups would be controlled by the exact same team of rehabilitation professionals but both groups were oblivious of each other's group exercise regime. The results of the initial study showed improvements with F-M and MS2 were similar between the two groups. However MS1 and MP showed significant differences between groups. The group involved with robotic therapy scored significantly higher than the group that was not using this technology.

The research went further than this and recalled patients two more times over a long period of time. The last time they were recalled was three years after first entering the rehabilitation center. Not all patients could be found, some patients had passed away and others had suffered a second stroke.

From the data obtained from the recalls it was seen that once again that F-M and MS2 were quite similar, while MP and MS1 were significantly different. The robot that was used during this rehabilitation process only concentrated on building control in the upper extremities. The motor control for wrist, fingers, and the upper limb subsection of the Fugl-Meyer had no inclusion in the robotic therapy administered.

This shows that the two sections which did receive robot-aided neuro-rehabilitation therapy were able to benefit from the therapy not only within the short term but also in the long term. The results found from this test are only a guide to the possibility of incorporating robotics used in rehabilitation. Extensive research must be continued within this field to confirm these results before a system such as this could be adopted within a rehabilitation program.

2.8 Short-Duration Robotic Therapy

Physical rehabilitation is usually done over a period of several months. One study [22] tested the effects of performing short duration robotic therapy for severe upper limb impairments. This study included 15 individuals who have suffered from a stroke ranging from a period of one to five years prior to being involved in this program. The patients would endure 18 sessions over a three week period. This involved two 1 hour sessions three times a week.

To be able to establish a clear benchmark to work from each patient was tested three times over a four week period prior to the beginning of the program. They were subjected to four different tests for establishing the benchmark. These four tests include, the Fugl-Meyer assessment, the motor power assessment, the Wolf motor function test and the stroke impact scale.

Once the program had started the patients were once again put through the four tests mentioned above as well as five robot-derived tests. These robot-derived tests are as follows, aiming error, mean speed, peak speed, mean:peak speed ratio and movement duration.

During the robot aided rehabilitation the patients would work on a robot called the InMotion2 (Interactive Motion Technologies) which was developed by Massachusetts Institute of Technology (MIT). During these sessions patients would complete three games. First attempt the robot would provide no assistance, after this first attempt an algorithm would determine how much assistance should be provided for the second attempt. A final attempt would then be performed with movement assistance at a constant level. After the three week period the patients were once again put through all nine tests.

From the tests performed it was found that there was an overall increase of muscle competency around five percent. The improvement can be seen more clearly below in figure 2.5. This figure shows the vast improvement that has been made since the patients first entered the program. Three months after the program had finished 13 of the 15 patients were recalled for further testing. It was found from the results that the average motor control capability was practically the same as when the patients had finished the program. This proves that even short duration rehabilitation will have a long term affect.

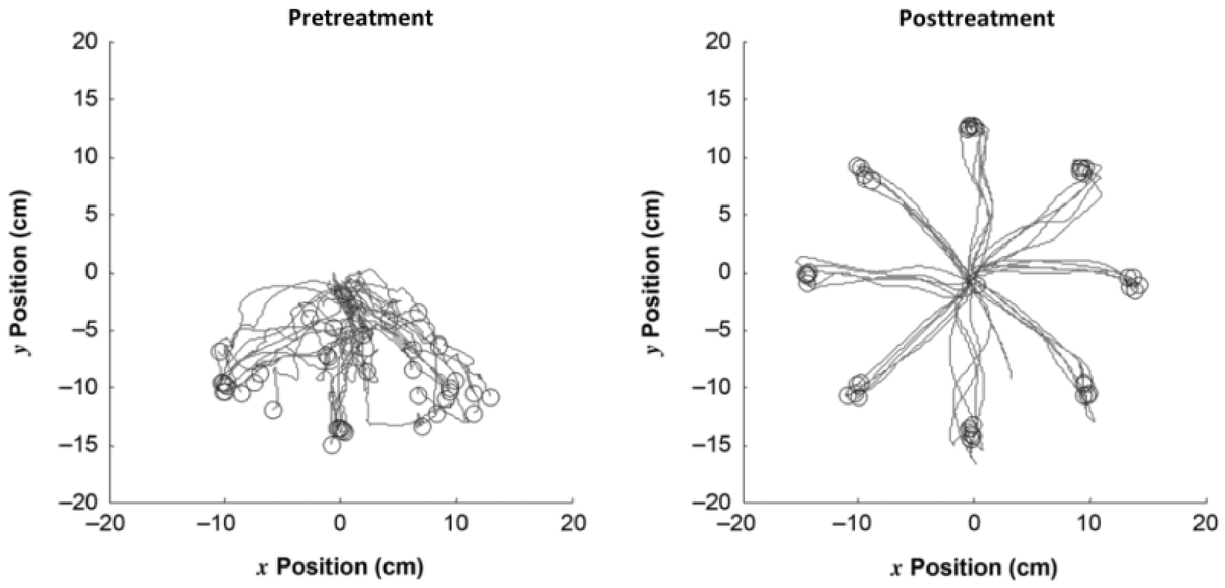


Figure 2.5: Pretreatment vs. Post treatment [22]

2.9 Planar Manipulandum Robot

There has been a difference of opinions by rehabilitation experts as to the best method to obtain the highest success rate. Should the rehabilitation process involve the patients paralyzed limb be given assistance to movements, or should the rehabilitation act in either a zero-force state or a resistive state. This study will discuss the effects of using active technology for training in a resistive state, after a period of resistive training the patients are then tested by the system when zero-force is supplied. First we will look at the design of the technology to give us a better understanding of the system.

The design of the system is best seen in figure 2.6. This system works using two brushed DC motors (rotational trajectory labeled 1 and 2 in figure 2.6) to provide the forces required, and rotational digital encoders to allow tracking [23]. DC motor 1 is located centrally feeding directly into the fixed end of the inside bar, DC motor 2 is also located centrally (i.e. DC motor 1 and DC motor 2 have the same rotational axis) but has an offset section that connects of the outer bar. This causes the location of point 2 to follow the inner circle of the diagram.

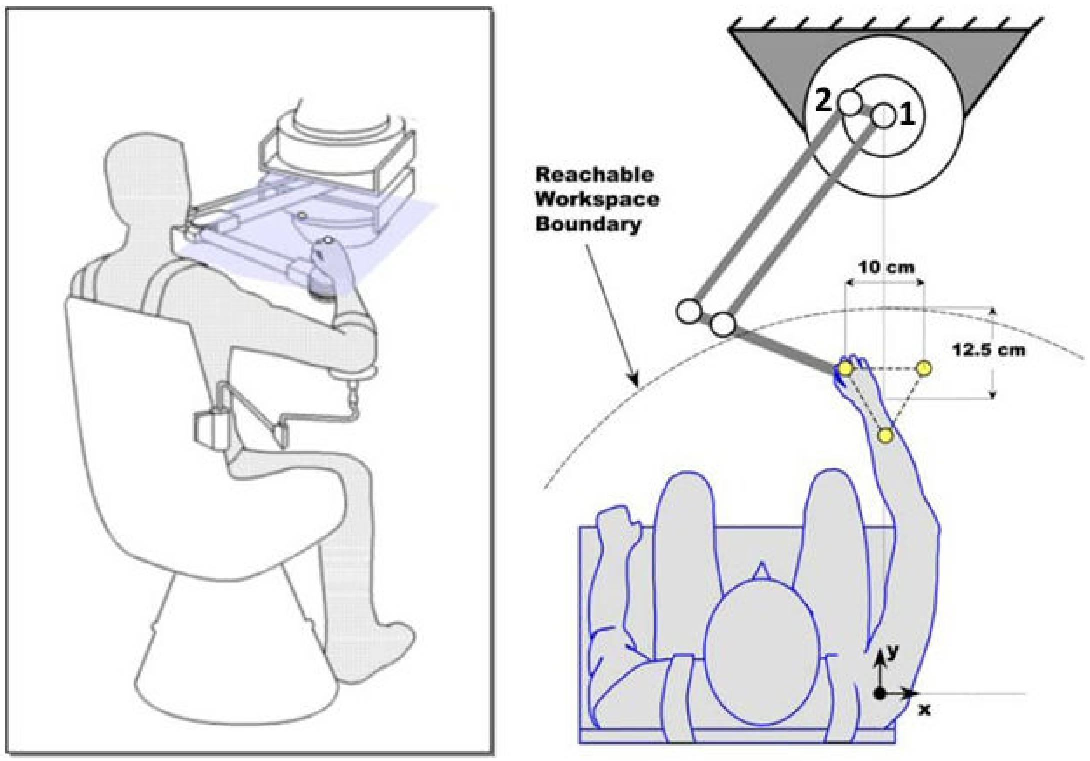


Figure 2.6: System Design [23]

This has the following effect on the system. As DC motor 1 moves clockwise/anticlockwise so does the bar it is attached to. For DC motor 2 the rotation has the same directional effect as DC motor 1. This system relies on a hinge point to achieve motion in the end bar which is attached to the handle. This hinge point is located nearest to the reachable work area boundary margin in figure 2.7. All the movements made by the two DC motors are best seen in this figure.

The study consisted of 27 patients all suffering from motor control problems. The patients each underwent the same program. This program consisted of a familiarization with the system (60 movements), an unperturbed baseline was then established (30 movements). Training and final training (372 movements) was then undertaken with an active system. The device was then changed to a zero-force system and the after effects (240 movements) were monitored. The final phase of the training consisted of the washout effect (120 movements). These five stages are best seen in figure 2.8. These diagrams consist of three line sections. The dotted red lines are the ideal trajectories, the thick blue lines are the average trajectories and thin grey lines are the individual trajectories.

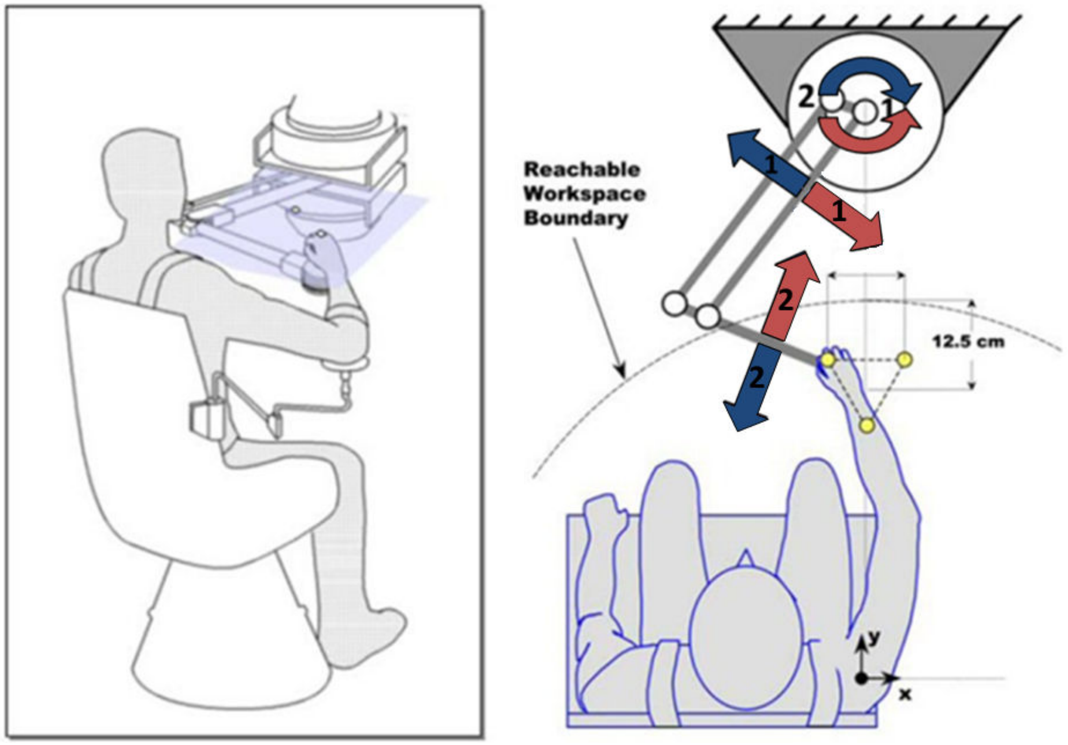


Figure 2.7: System Movements [23] (image has been modified)

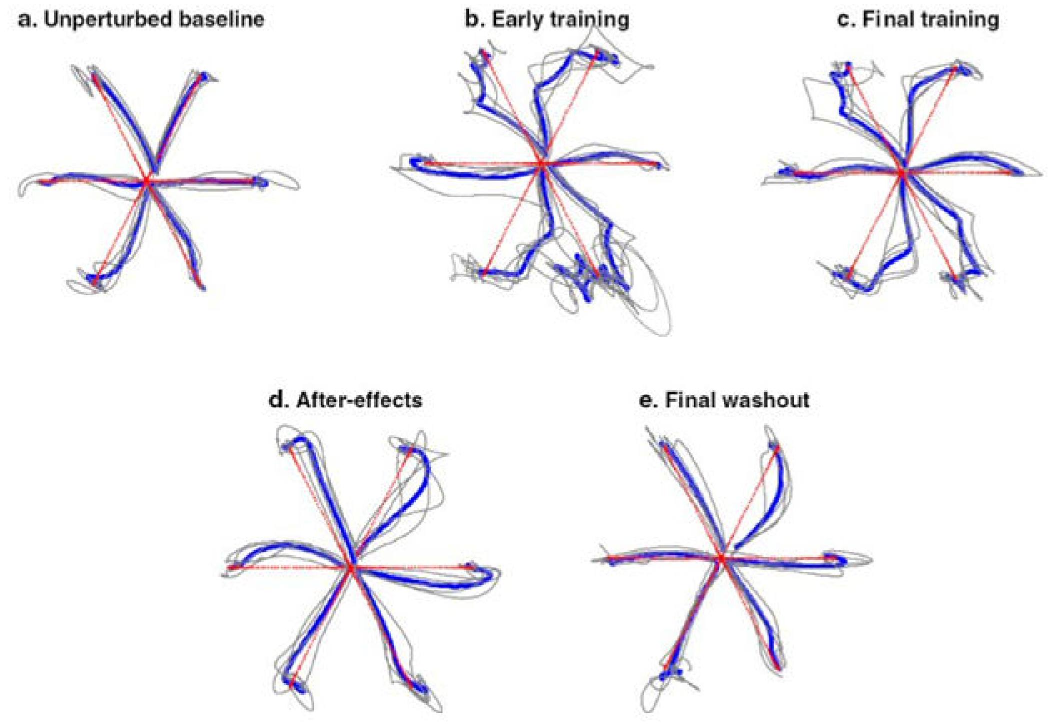


Figure 2.8: Overall Results [23]

From the test conducted it is obvious that stroke patients are able to retain their ability to adapt their arm movements to oppose external forces introduced. This supplies strong evidence that the use of resistance training during rehabilitation is beneficial for the patient.

2.10 Conclusion

The use of robotics in the field of physical rehabilitation should not be a feared integration by staff or patients but more of a welcomed one. This new technology should not be used to reduce the number of trained staff that are currently practicing physiotherapy but should be used in correlation with them to provide the patient with more rehabilitation then time would normally permit. One physiotherapist would be able to handle a number of patients at a single time, each of them receiving the same quality that they would normally expect from having a personal physiotherapist in a one on one situation. This coupled with the potential to provide a more successful rehabilitation program would be a large step forward in the field of physical therapy and therefore the rehabilitation of the patient. As with any rehabilitation program if you can rehabilitate a person faster and have them self sufficient, you are in a position to assist more people requiring treatment.

Several issues have been raised in this chapter that is crucial to better understanding the rehabilitation of stroke patients. One of the most important issues raised was that stroke recovery can occur decades after the stroke has occurred. Although it is still crucial for rehabilitation to be performed as soon as possible following a stroke while the muscles and tendons are still in a healthy state.

The most important discovery from the research is that there is significant evidence that patients who have been involved with robot-aided neuro-rehabilitation have ultimately gained more control over their effected limbs compared to those who have undergone a standard rehabilitation. The three goals aimed to be achieved by rehabilitation are improving a patient’s perception, coordination and muscle strength. By improving in these areas a patient should be able to continue to live a life without disability. The following section will discuss the conceptual ideas created to incorporate all requirements into a mechanical system.

Chapter 3 Conceptual Designs

3.1 Introduction

Taking into account the many factors derived from previous chapters I have developed several designs which would be able to be manufactured to aid in the physical rehabilitation of stroke patients. Some of these ideas which seemed practical at first turned out to be complex, expensive or just not feasible upon further investigation. Some of these ideas helped lead to the evolution of the design that I decided to proceed with. In the proceeding sections throughout this chapter I will explain the ideas which have worked and those which have not proven feasible to continue with.

The system being designed in this section aims to act as a rehabilitation tool to be used in coordination with rehabilitation from a physical therapist. The patients this rehabilitation technology aims to assist are patients who suffer hemiparesis of the upper extremities resulting from a stroke.

3.2 Design Requirements

Before the system could be designed we must first know the range-of-motion that would be required for the exercises. This research is titled, design and build a computer based system to assess and exercise the arm movements of stroke patients. From this it was decided that the system would work only within two planes (x-y planes). This specification is crucial for the system design to continue.

During physical therapy the exercises that are practiced are limited to an area to avoid straining the patient's muscles, this area is referred to as the range-of-motion and differs for each patient. In conjunction with the range-of-motion required for an individual patient exercises will also only be practiced within an isolated area. A prototype developed during the research allowed a physically able person to explore different ranges-of-motion that would maximize physical exercise while not being over strenuous. By performing this physical experiment it was found that a maximum area which would be required for the device to operate in would be approximately 600 millimeters by 800 millimeters. This test was performed by an average size male adult. The area of operation is shown in figure 3.1.

It can be seen from this figure that the patient would be performing rehabilitation on their right arm. A left handed patient would require the mirror image of this figure. In rare situations the rehabilitation system may require a larger area for operation then mentioned above. The possibility of this occurring should be incorporated into the system to allow the device to be operated by any user possible.

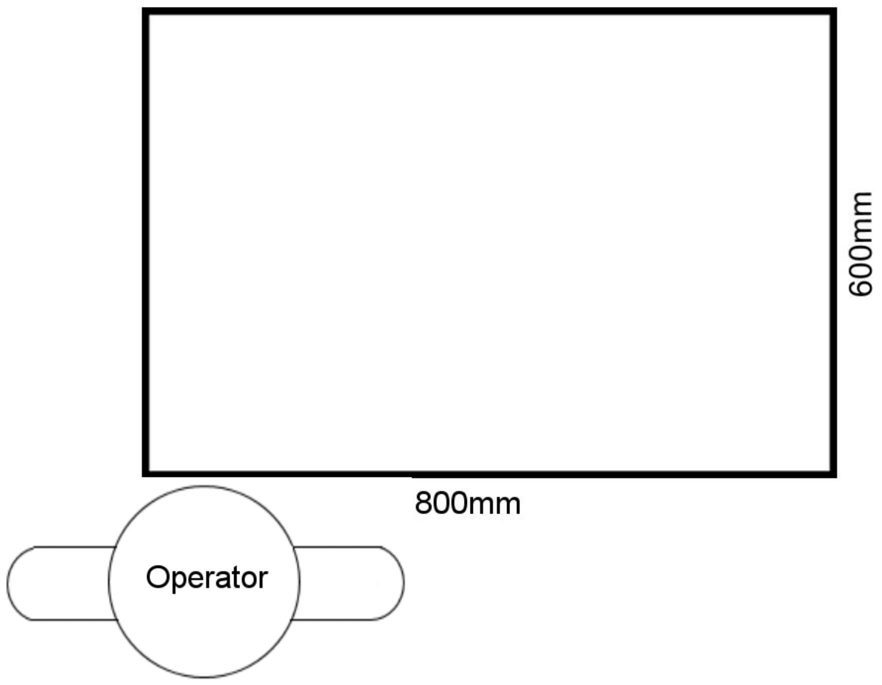


Figure 3.1: Work Area

It is crucial for the system to be able to exercise as many different muscles during its operation. Working different muscles encourages new muscle development and allows the retraining of the brain to be able to operate these muscles. An appropriate way to achieve this is by incorporating into the system changes in direction. A simplified example of a system which could achieve this is best described by figure 3.2.

The triangular pattern shown in this figure is very basic but still allows patients to exercise different muscles. The patient would first start at P1 (point one), the patient would then move from P1 to P2. In doing this the patient has used several muscles, some muscles to accelerate and decelerate their arm and the rehabilitation device, as well as using muscles to control the movement to operate around the idealistic straight line between these two points. The patient would continue this pattern moving from P2 to P3 and then returning to P1.

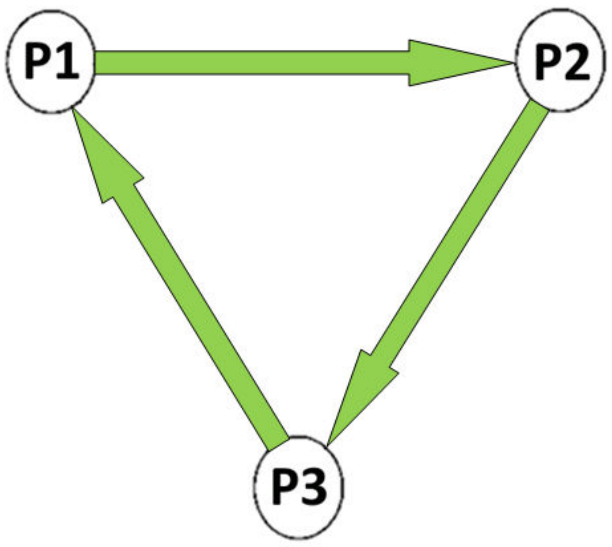


Figure 3.2: Simple Exercise Pattern

A more advanced exercise pattern can be seen in figure 3.3, this pattern requires the full range of muscle groups for planar movements. The patient would be required to start at the centre location and move to P1. Once the target has been achieved it is required the patient return to the centre point before moving onto P2. This will be repeated until all six targets have been reached. This pattern is taken from the work done by James Patton.

The system should also incorporate complex movements for the operator to achieve. Complex movements are ones that would require the use of several different muscles to work in unison with one another to achieve coordinated movements. An example of a complex movement is buttoning a tee-shirt, this requires coordination and precision of several different muscle groups.

The last system requirement is that the system must be able to support passive and active movements. Passive movements are movements that require no physical assistance whereas active movements do require physical assistance.

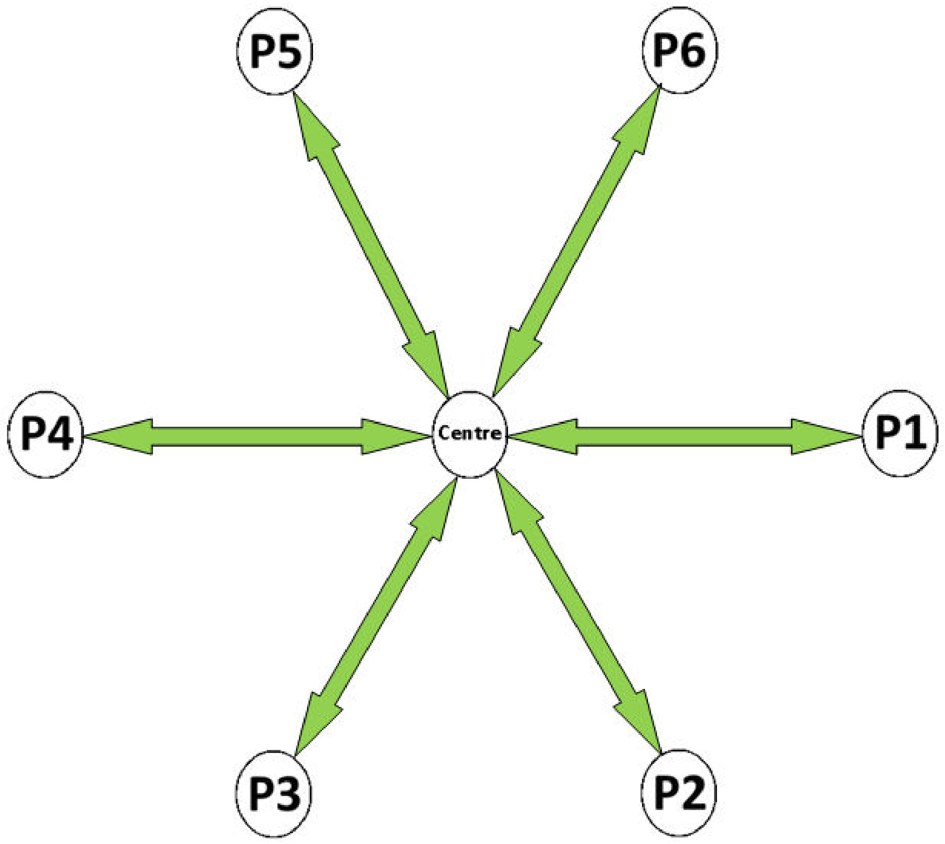


Figure 3.3: More Advanced Exercise Pattern

3.3 Conceptual Designs

The following section discusses the conceptual designs created during the initial stages of brainstorming.

3.3.1 Design 1:

During the development of my design I tried to completely change the general design that is common with current technologies. This design simplified many problems associated with previous designs, but had one large flaw.

An idea of what I wanted to create can be seen in figure 3.4 and figure 3.5. The system would appear to be half a sphere when sitting on a bench. The patient would simply rest their hand onto of the device and move it to the required locations, this eliminated any joint or weight issues. The device would be mostly made of plastic. This keeps the system lightweight and plays a vital role in the systems operation.

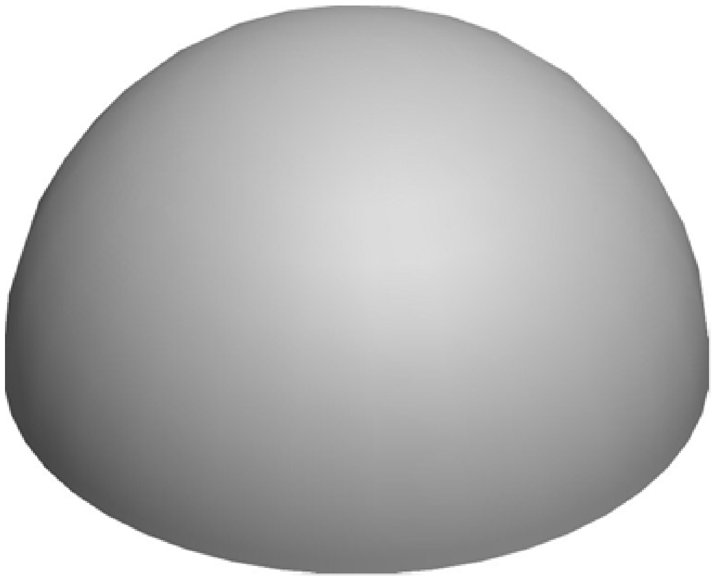


Figure 3.4: View From Above

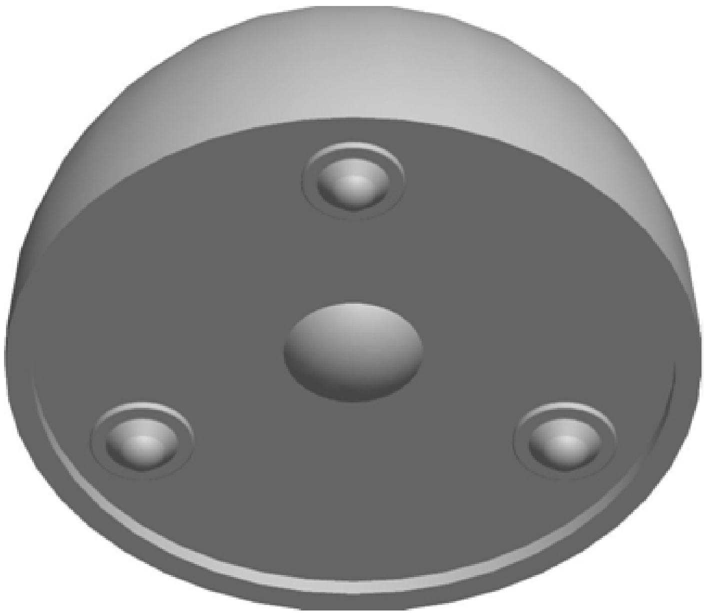


Figure 3.5: View From Underneath

Underneath this design would be several roller balls to support the system off the bench thus creating a freely moving system operating within two dimensions. The roller balls play one more crucial role into the operation of this system. The three outer roller balls would be designed as shown in figure 3.6. Each roller ball would contain a spring, pressure brake and the roller ball.

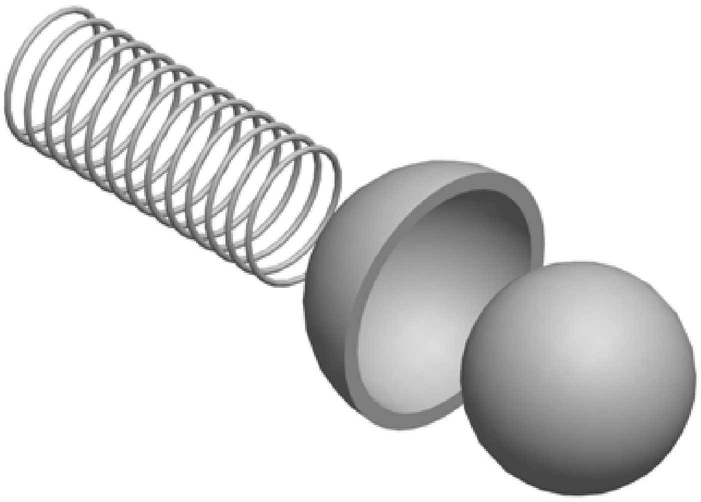


Figure 3.6: System Mechanics

The device would operate on a special bench which has electromagnets incorporated into it. When a freely moving system is required the electro-magnets are disabled. When resistance is required the electromagnets are enabled. A metal ring attached to the bottom would pull the plastic body of the device downwards and enable the pressure brake. This would create resistance for the system to roll freely. The rolling resistance is proportional to the amount of electromagnetic force produced.

The downfall of this design was simple, the system was able to provided resistance to movement but could not provide assistance to movements. As explained in the previous chapter active movements are crucial to the redevelopment of motor control.

Advantages:

- Has complete freedom in two dimensions
- Very cheap to build
- Simple design
- Very light

Disadvantages:

- Does not support active movements
- Electromagnets could affect computers
- Magnetic waves could be unhealthy

- Tracking method requires roof mounted camera described in chapter 4

3.3.2 Design 2:

This system requires only two degrees of freedom. One degree of freedom is located at the fixed base, this is a point of rotation. The other degree of freedom is an axial side along the shaft of the arm. This can be seen in figure 3.7. The combination of these two degrees of freedom allows two dimensional movements.

This system could either float in free air or slide across a bench in the same fashion as the previous design. The preferable setup of this design is to freely float in the air. This way the device would be easily incorporated into a variety of different situations. For example it could be adapted to suit a wheel chair.

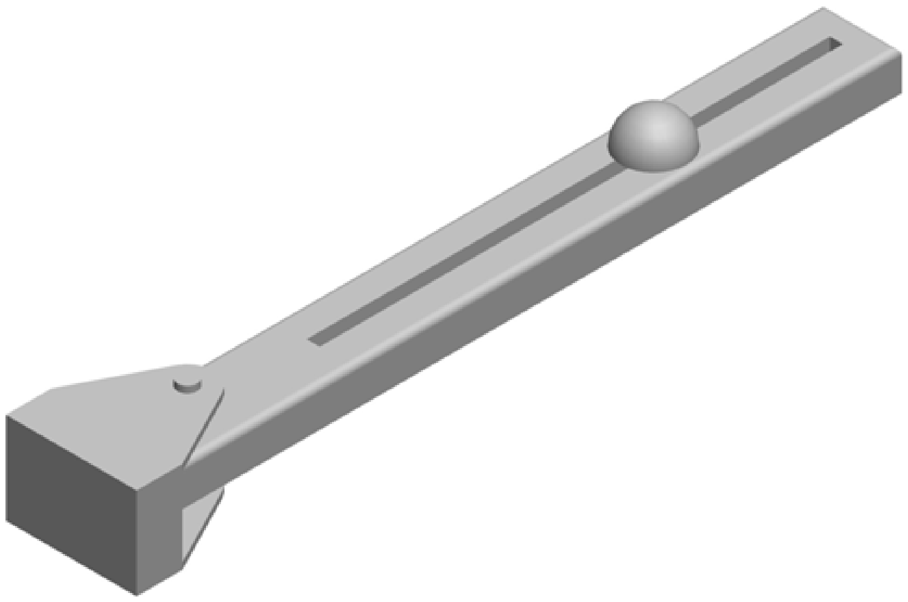


Figure 3.7: Slide and Hinge Arrangement

This design would require significant labor and materials compared to the first design. Using a slide in the system introduces some problems which could become expensive to overcome. One such problem is how to incorporate a mechanical system into the slide to allow active movements which would still allow the system to be freely moving in a passive state. Overall this system has some advantages over the previous system but also introduces some new problems.

Advantages:

- Does not suffer from joint locking
- Can be attached to a wheelchair

Disadvantages:

- Overall system is heavier
- System is more complicated for smooth operation
- Limited methods to make system active
- System would be more expensive to manufacture

3.3.3 Design 3:

The third conceptual design is a system similar to many current designs that are currently being developed. The system has two degrees of freedom, the first point of freedom is positioned at the fixed end of the device. The other point of freedom is located centrally between the fixed end point and the handle of the device. These two points of freedom allow a freely moving system in a two dimensional plane. To achieve the required 800 millimetre by 600 millimetre working area specified earlier the distance between each center point of rotation is 700 millimetres. The required work area can be achieved using smaller separating lengths but by reducing these lengths problems are introduced into the system.

By using arm lengths shorter than what I have specified there are two major problems. The first of these problems can be seen in figure 3.8. When the handle approaches the fixed point the system becomes increasingly heavy to move. When this occurs and the patient attempts to move the handle in one of the directions shown in the diagram, most of the force applied by the patient is exerted into the joint in a shear manner rather than actually causing rotation.

The second problem as shown in figure 3.9 is when the handle, middle point and fixed end point all align causing joint locking. At this point it becomes difficult and uncomfortable to operate the device. Electrical wiring will run around this joint and could be damaged if joint locking occurred. It can be seen from these two diagrams that the optimum working area would be located away from these two possibilities. For an optimum working area this arm will only work within one quadrant of possible ranges.

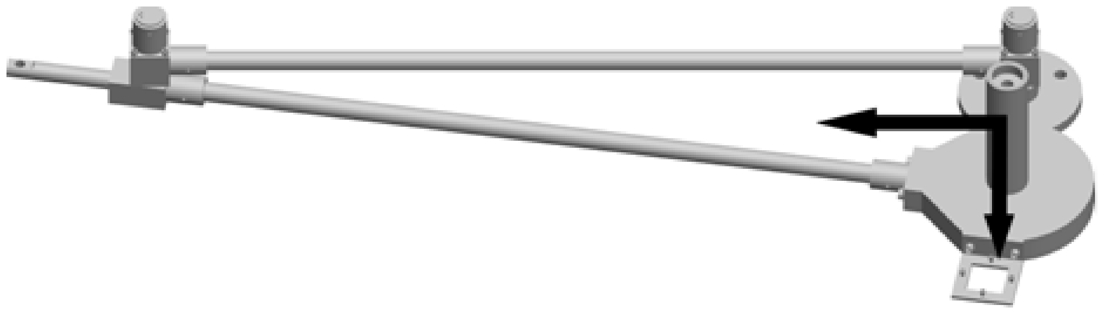


Figure 3.8: Avoided Work Area

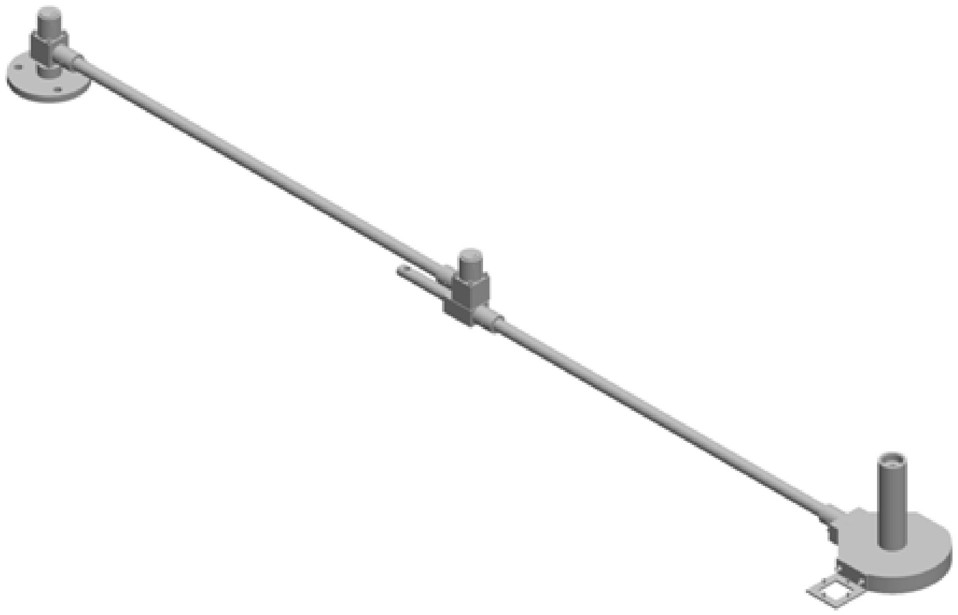


Figure 3.9: Joint Locking

It can be seen from the above diagrams that the handle of the device is quite large and cumbersome. This has been designed so large for a reason, the operator will not be able to support the weight of their own arm thus their hand will have to be strapped to this device. Therefore this large base will give ample room for a large hand to sit comfortably. It was difficult to come to an optimum size so I opted for a large size to try and achieve a balance for a wide variety of sizes.

An advantage of this system is that when introducing mechanical control to simulate active movements many mechanical devices can be easily adapted to do this. Active control will be covered in the following chapter.

A brief summary of the advantages and disadvantages are as follows.

Advantages:

- The system is lightweight
- Relatively inexpensive to build
- Design is simple
- Handles various mechanical systems for active assistance
- Provides a large, freely moving work area

Disadvantages:

- Optimum working range is only in one quadrant
- Crosshairs could become difficult to see
- Requires crosshairs to be physically moved to optimize usages for left hand/right hand users

This design was used for further investigations into accomplishing the other functions required from the system.

3.4 Assistance/Resistance to Movements

Information gained from chapter two discussed several requirements for the successful rehabilitation of a stroke patient. The most significant requirement of the entire system is based around the system being able to support active movements. Active movements refer to movements that are assisted by an external body (either human or mechanical). These active movements are provided in one of two ways, either assisted movements or resisted movements. This following section describes each technology investigated in detail, and discusses the benefits of each system as well as the problems associated with it.

3.4.1 Linear Pneumatics

There are several different styles of linear pneumatics but they all follow the same general structure. Shown in figure 3.10 is the basic concept of how a pneumatic operates. Air is forced into one end of pneumatic shifting the rod in the relative direction. Major differences in pneumatic designs are the number of rods in the pneumatic, operational air pressure and how the rod/rods are manipulated. Pneumatics support either bi-directional air flow or single direction air flow with a spring return. Operating pneumatics in a controlled manner requires several components which include solenoid valves, a pressure regulator, air reservoir and many other small connecting sections and drivers which can quickly become expensive.

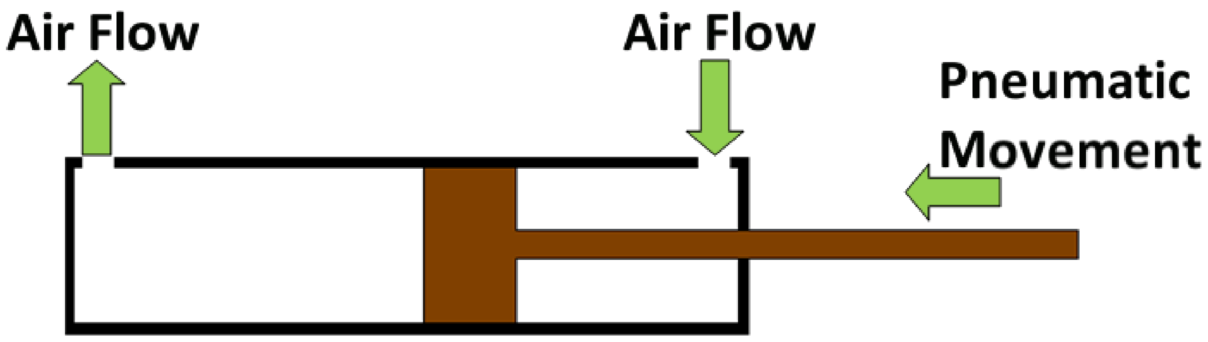


Figure 3.10: Linear Pneumatic Diagram

Linear pneumatics seemed to be an option that would allow simple integration into the chosen design but upon further investigation conflicted with my initial opinion. To implement linear pneumatics into the design I extended the joint past the point of rotation. This can best be seen in figure 3.11 and figure 3.12.

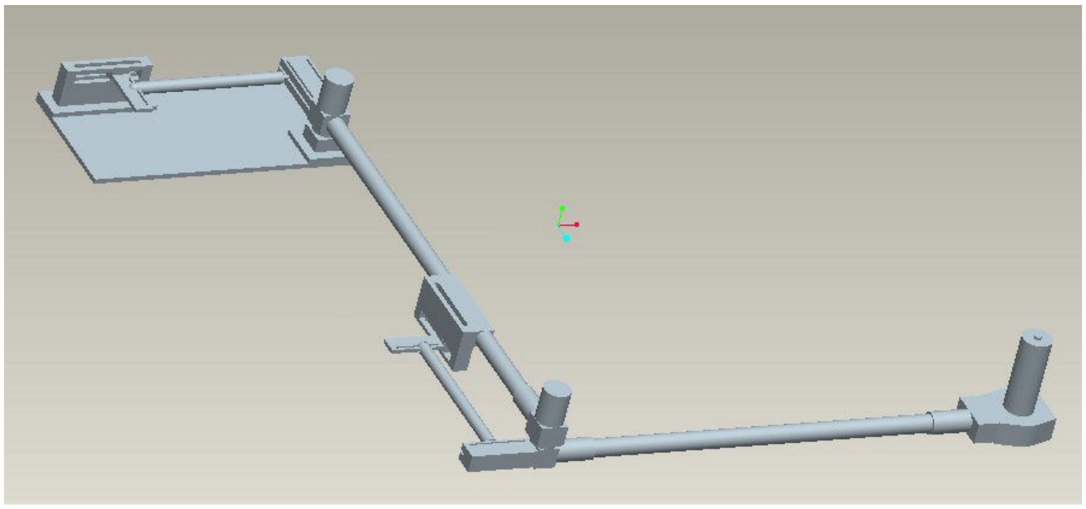


Figure 3.11: Initial Pneumatic System

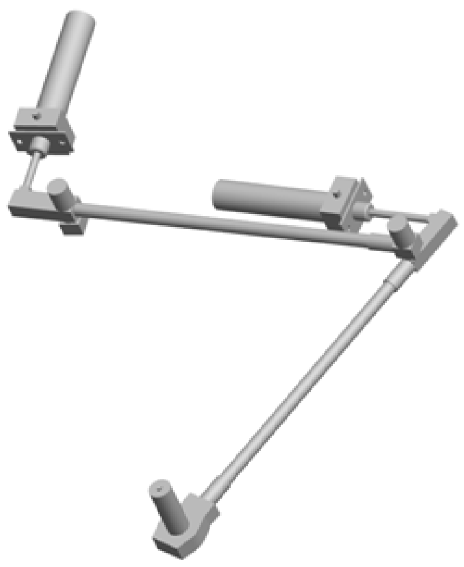


Figure 3.12: Linear Pneumatic Implementation

This extended section allows a linear pneumatic to be attached at this point. The other fixed point of the pneumatic required being on a swivel point to allow the transfer of an axial linear motion to be transferred into a rotational motion. The pneumatic would require being mounted close to the joint otherwise fouling could occur between the pneumatic and the point of rotation. This additional bracing would create weight issues on the middle joint. Further information regarding the problems and benefits of linear pneumatic systems are listed below.

Advantages:

- Simple to control
- Easily integrated into the device

Disadvantages:

- Requires air line
- Requires bulky supports on middle and end joint
- Are expensive
- Non-linear rotation

3.4.2 Rotary Pneumatics

A rotary pneumatic differs from a linear pneumatic by how the internal components are arranged. The two types of rotary pneumatics available are a turbine style pneumatic which involves pressurized air crossing a turbine causing rotation of the shaft. The other style available is a rack and pinion design. The rack and pinion design was the system investigated, this design involves injecting air onto the end of the toothed rack causing its position to shift along a track. A centrally located pinion rotates relative to the rack movements. This can be seen in figure 3.13. In a single rotary pneumatic can be up to four racks.

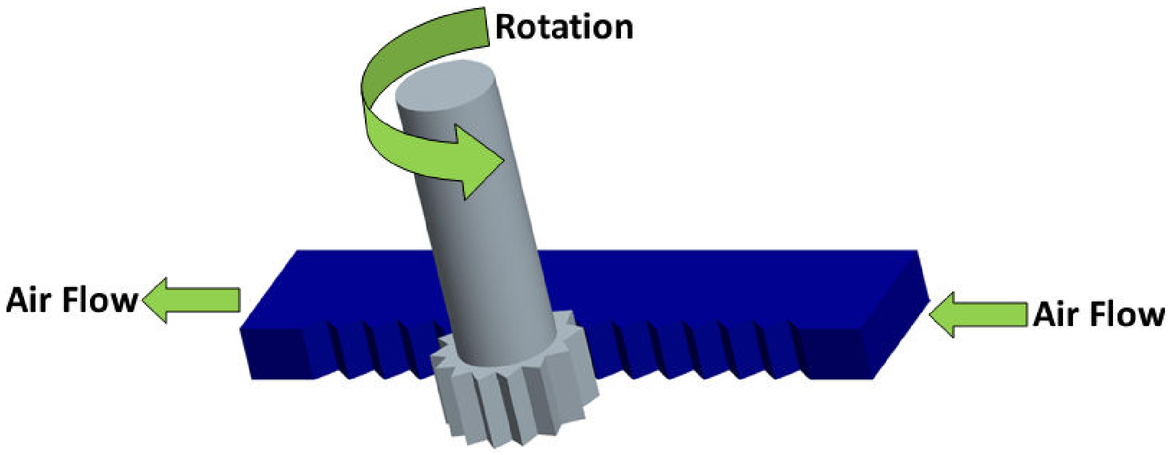


Figure 3.13: Single Rack and Pinion

Rotary Pneumatics could easily be implemented into the system by simple joint modifications. The joint modifications that would be required are best seen in figure 3.14.

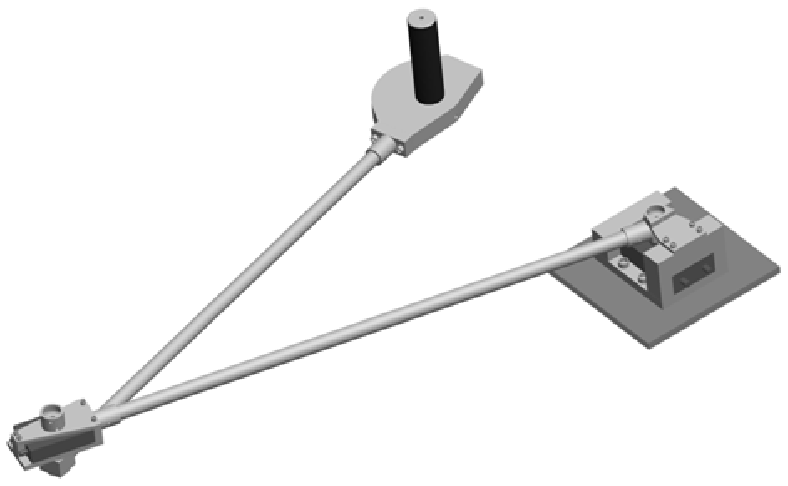


Figure 3.14: Rotary Pneumatic Implementation

Rotary pneumatics have one major benefit compared with a linear pneumatic. That benefit is that rotary pneumatics could achieve a linear rotation. The only problem was the speed of rotation. From my research the slowest acting rotary pneumatic was 50 degrees per second. This is obviously too fast for a disabled person let alone and able bodied person. Further points are summarized below.

Advantages:

- Linear rotation
- Easily controlled
- Easily implemented into device

Disadvantages:

- Rotation speeds (minimum of 50 degrees per second)
- Expensive
- Require airline

3.4.3 Air Muscles

Air muscles have several advantages over the previous systems investigated. These advantages include:

- Low cost
- Light weight
- Simple to implement

Besides these advantages there is one drawback with using air muscles. Just like a real muscle, air muscles can only provide one force, which is contraction. The implementation of air muscles would require one on the outside and one on the inside of each joint. This is not that big of a problem but might clutter the arm and also act as a visual deterrent for users. This arrangement can be seen in figure 3.15. Controlling an air muscle is done in the same manner as a single direction linear pneumatic.

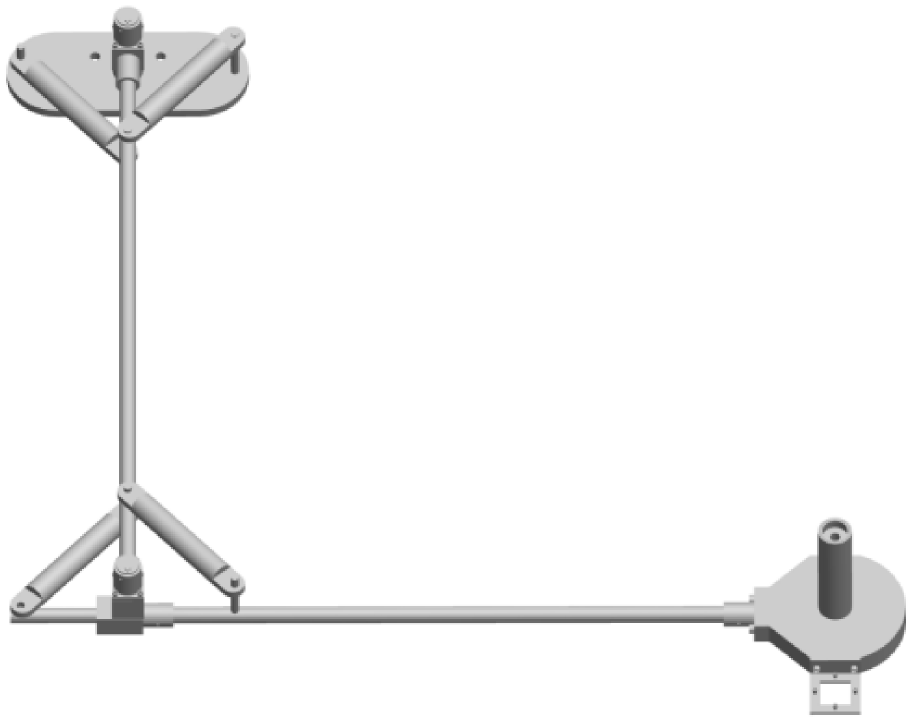


Figure 3.15: Air Muscle Implementation

With the three systems mentioned so far there is one large problem, this is the average person does not have compressed air available within their house, this problem makes these technologies inconvenient for household use. It also introduces the requirement of an air compressor being required, this increase the cost of the overall system.

Advantages:

- Cheap
- Lightweight
- Easily implemented
- Easily controlled

Disadvantages:

- Requires two per joint
- Requires airline

3.4.4 DC Motors

The implementation of DC motors allows several advantages over other systems. First of all it fulfills one of the major requirements, this is for the device to be a technology of convenience. A technology that is able to be used at home with no inconvenient requirements such as an air compressor. The DC motor will be able to be introduced into the arm system in a similar fashion as the linear pneumatics were. One major difference being that the DC motors will be able to be fixed at a location that will not contribute to system weight. It will simply have an extension bar running from the motor to the required joint. This setup can be seen in figure 3.16.

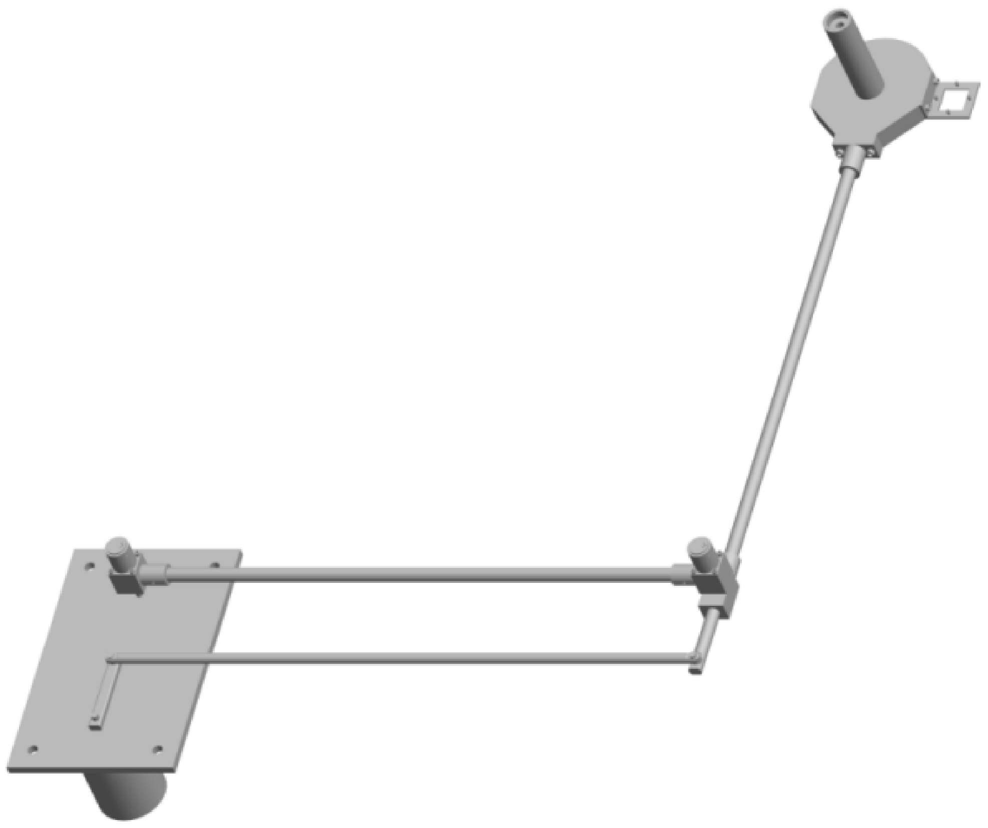


Figure 3.16: DC Motor Implementation

Advantages:

- Can be easily implemented
- Reasonably price
- Runs of electricity
- Quiet

Disadvantages:

- Base becomes cluttered with motors
- DC motors are large and cumbersome

3.5 Tracking Technology

It is essential for a stroke patient who is undergoing physical rehabilitation to have a personal program developed to suit their own requirements to achieve the best results. Physical rehabilitation performed solely with the interaction of a physical therapist relies on the memory and ability of the therapist to determine the extent of any improvements made by the patient and to redesign their exercise routine in accordance with this information. By implementing technology we are now able to record and log very accurately every movement made by the patient. This allows for less interaction with a physical therapist with an overall better result. There are several technologies that I investigated to achieve this movement tracking. These technologies are described in the following sections.

3.5.1 Roof mounted camera

This system is a machine vision based system. A camera would be mounted directly above the working area where every part of the working area is visible. Located on the handle of the technology in development would be a green spot. The camera would use a low pass filter to remove the other colours allowing tracing the location of the green spot to take place. This system is quite sophisticated and a simpler system could be incorporated to accomplish the task. A disadvantage of this system is the complication of a camera being required to be mounted above the work area. Thus this system would not be feasible for home use due to the complications of transporting and setting up the equipment to monitor the patient receiving treatment.

3.5.2 Bench Mounted Camera

This system is almost identical to the previous system but differs with the camera mounting position. In this situation the camera would be mounted underneath the device, requiring a specially designed workbench with a glass top. This is an expensive and once again over complicated idea to achieve this task.

3.5.3 Optical Mouse

The simplest of all options would be to incorporate an already successful technology such as an optical mouse into my designs. The tracking of the mouse would be accomplished easily using functions which preexist within the VB.NET framework. There are however some difficulties in using an optical mouse.

Since the computer would recognize the disassembled optical mouse used in the handle of the technology as a standard mouse it could cause confusion between that mouse and the mouse associated with the general operation of the computer.

3.5.4 Potentiometer

The use of potentiometers was the final technology investigated into suitable methods of tracking. A potentiometer is simply a variable resistor, as the top section of the potentiometer rotates in an opposite direction with the bottom section resistance through the system changes. In our case we are then able to determine from the change of resistance a relative angle. This can be seen clearly in figure 3.17 below.

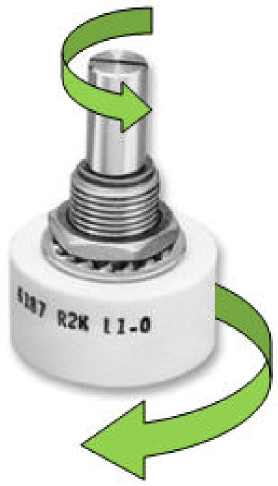


Figure 3.17: Potentiometer Operation [24]

3.6 Chosen Design

From the designs and technologies discussed during this chapter I have reached a final design. This design is best described in figure 3.18. The design structure is as follows:

1. DC motor used to control angle changes around point 4.
2. DC motor used to control angle changes around point 3.
3. Housing for a single turn precision potentiometer used for tracking.
4. Housing for a single turn precision potentiometer used for tracking.

5. Removable section to enable the use of cross hairs. This is interchangeable between both sides of the handle.

6. Handle with an inbuilt button located on the top.

Full technical information about the precision potentiometers used can be seen in Appendix D.

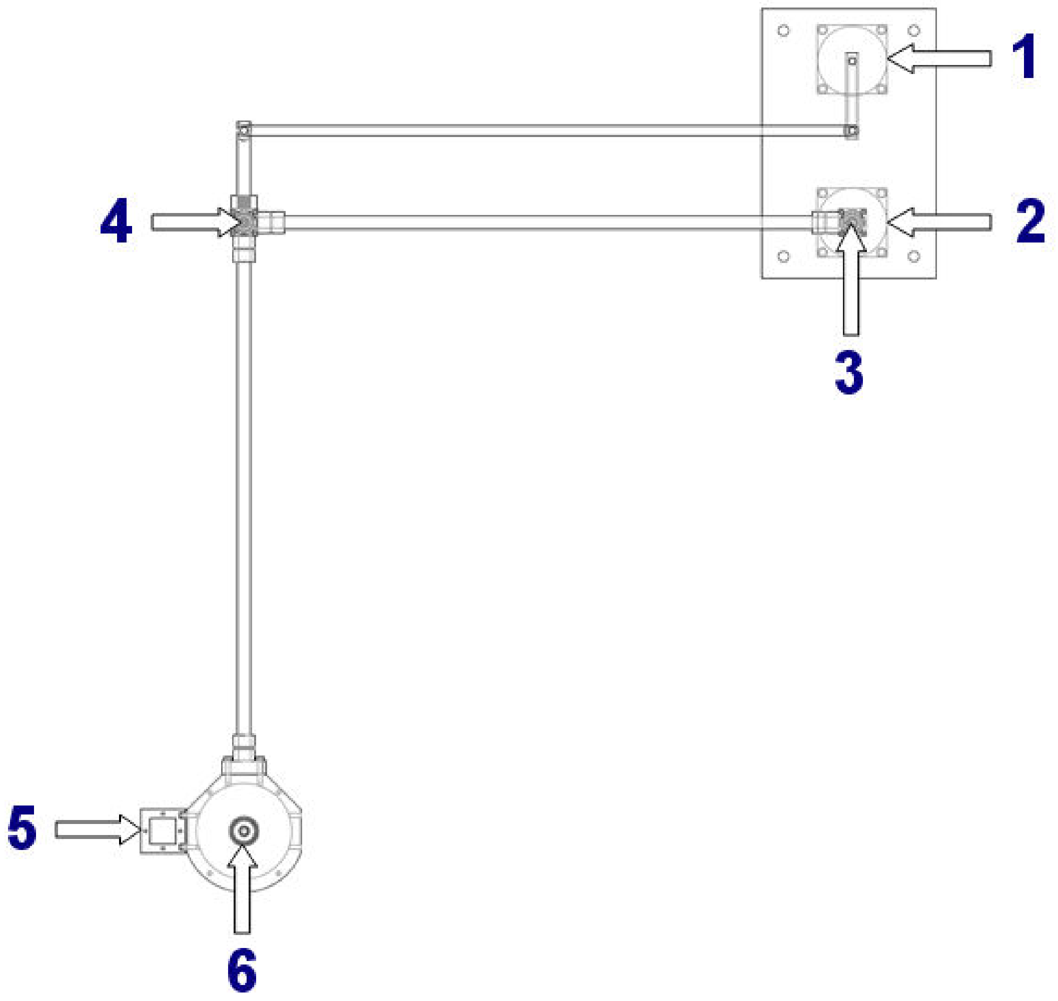


Figure 3.18: Chosen Design

3.7 Safety

Rehabilitation technology could easily cause more damage to the patient if it is not correctly controlled. During the paralysis experienced after a stroke many parts of the body change. One such section is tendons, after a period of time tendons shrink causing the arm to become locked. If a mechanical device which a patient is strapped to were to malfunction or pull with a dangerous force body parts such as tendon could be torn and require surgery to fix.

Therefore there will be many safety aspects introduced into the system to remove as many of the adherent dangers as possible. These safety aspects are as follows:

- Limit motor torque

  The best way to prevent bodily damage from occurring is by restricting the output force that the DC motors are physically able to produce during an active system. The force required to move a patients arm suspended above this device would be quite small therefore the motors required to provide active assistance would be unlikely to be able to cause significant bodily damage.

  Not all the patients will have the same requirements (one patient might weigh 45kg but then be followed by someone who weighs 90kg) therefore the supply voltage provided to the motors will be able to be physically altered to control the output torque. The control of the supply voltage will only be able to be altered by trained professionals. As a second measure to prevent an incorrect supply voltage from being provided each patient would sign into the program which would load up their profile. Contained in this profile would be the required supply voltage, if this did not correlate with the supply currently in use, the program would not commence.

- Emergency switch

  At all times of operation a patient would be required to place their hand (from the non-paralyzed side of their body) over an emergency switch. This would allow the patient to disable the active assistance instantly if ever required.

- Easily Remove hand from strapping

  During normal operation of the device the patients hand will be physically attached to the handle of the device. A patient with no motor control over their fingers would not be able to hold onto the handle otherwise. The strapping used for hand restraint should be designed to allow their hand to be quickly removed. E.g. a Velcro strap. This will avoid the patient becoming physically attached to the device in an emergency situation.

- Proper training

  The most important safety factor is proper training. Both the physiotherapist and patients should be thoroughly trained to understand all the programs functionality and emergency procedures before any work is commenced.

The system under development will have many moving parts. This introduces the possibility of limbs or clothing to become trapped or tangled which could lead to further injuries or other dangers. The device will have any dangerous sections covered to eliminate the chance of something becoming caught and also remove pinch points.

3.8 Conclusion

There are many possible ways to create a system to fulfill the requirements specified for stroke rehabilitation. Many of these quickly become expensive and not feasible for home use. To create a system that is simple, has low manufacturing costs and is able to used anywhere will allow this kind of technology to be easily implemented into a rehabilitation program.

This past chapter has discussed many technologies which could be introduced into the chosen design to allow active controlled to be implemented. From these technologies it was found that the most feasible system to introduce was a system involving two DC motors. By using two DC motors we would be able to keep the system simple, cheap, quiet and versatile.

To achieve motion tracking it was decided that two precision single turn potentiometers would be introduced into the rotating joints. The resistance change from the potentiometers would be measured and an angle derived relative to this change.

During the development of the technology safety was the highest priority. Systems were chosen to avoid unnecessary dangers from existing. Many safety aspects were introduced into the completed system to allow a safe operation during exercise periods. The following section discusses the prototype created, development board technology used and the GUI created to be used in coordination with the stroke technology.

Chapter 4
Prototype Development & GUI

4.1 Introduction

To establish an accurate scale for the chosen device it would be constructed to allow an optimal range-of-motion. A simple prototype of the device would be constructed before any manufacturing occurs. This following chapter discusses the prototype and final material selection.

This module introduces the development board used to enable communication between the device and a standard PC with an RS232 connection. It details the pin arrangement and the technologies used for conversions integrated into the chipset.

The development and structure of the GUI to be used in coordination with the manufactured product will be broken down into sections detailing how the program has accomplished the task and allows users to have versatility while operating the device.

4.2 Prototype

The development of a prototype is crucial step towards defining standards that the system will be able to operate within. I gathered several ideas for the construction of the joints from a preexisting three dimensional device. From this I was able to construct a very simple prototype which can be seen in figure 4.1.

There were several system criteria which I derived from this prototype these include which quadrant the system should operate in, where the optimum position of the base should be relative to the operating area and most importantly I was able to derive an appropriate distance between the center points of rotation as being 700mm for optimum operation.

By using this model I was able to quickly and easily modify parts of my design and be able to feel how the operation of the device would feel like with different physical characteristics.

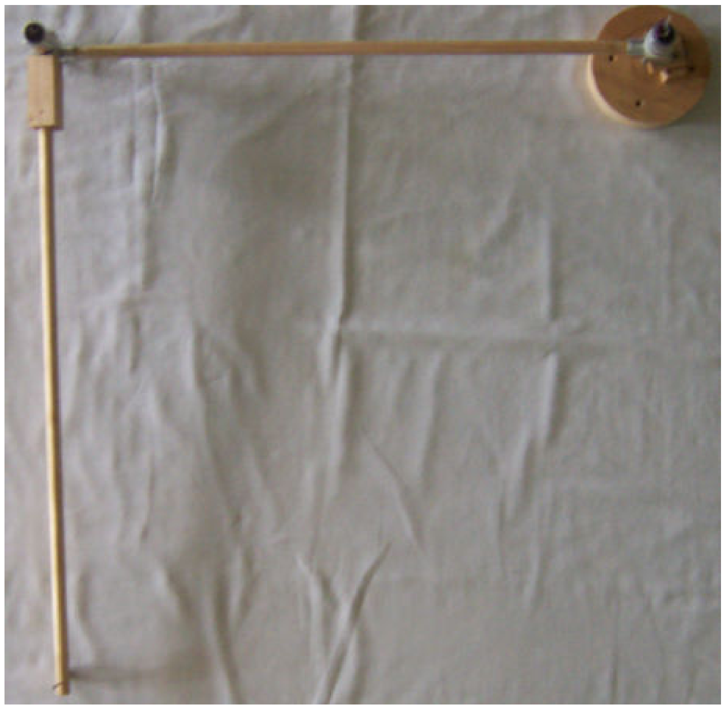


Figure 4.1: Wooden Model Prototype

4.3 Materials

In choosing an appropriate material for the device to be manufactured from there was several factors that had to be taken into account. These factors related to the user’s ability to move their effected limbs, and also for allowing further development of the device to introduce active movements.

It was decided that the first manufactured stroke rehabilitation device would be manufactured entirely from aluminum. This material was selected based upon its known properties of being a strong material while still remaining light weight. It was known prior to manufacturing that the overall system weight would be significantly more than a patient undergoing physical rehabilitation would be able to move. After consideration it was decided that the systems overall weight would not play a major role once the technology had been finished being developed.

The reason the weight factor was ignored comes back to the fact that the technology is required to be developed further before being tested. Once the system has further components added to provide assistance and resistance the weight of the system will no longer be a factor. One option that would be integrated into the program would allow the active system to be disabled providing a free flowing system. This would of course be used as an in between step from providing assistance to providing resistance.

4.4 Redesign for manufacture

When developing the initial design, I designed components based around the most suitable parts available rather than common components which would be available for use during construction. Initial designs also tended to be over complicated and would lead to longer build times resulting in higher cost for construction after consulting the machinist I was able to gain a better understanding of machining techniques and requirements. Taking all these factors into account I made many minor modifications to simplify the overall design making the construction simpler, easier and cheaper to construct.

4.5 Tracking Data Capture

As discussed in the previous section it was decided that the technology that would be integrated into the system to accomplish the tracking movements would be potentiometers. To better understand the implementation of potentiometers into my design it is best seen as an exploded view shown in figure 4.2. The shaft of the potentiometer fits into the end of the locking pin, this is held firmly in place with a grub screw. The body of the potentiometer slides into the top section of the housing and once again is held firmly in place by using 3 grub screws. This allows the potentiometer to be freely rotated when the middle section of the device rotates.

When the handle of the device is moved either one or both potentiometers will be rotated. This will result in an analogue resistance change to be output from which an angle can be derived. To be able to convert this analogue signal into a signal that a computer is able to understand (a digital signal) I will be using a development board (LPC-MT-2138) which has several integrated ADC’s.

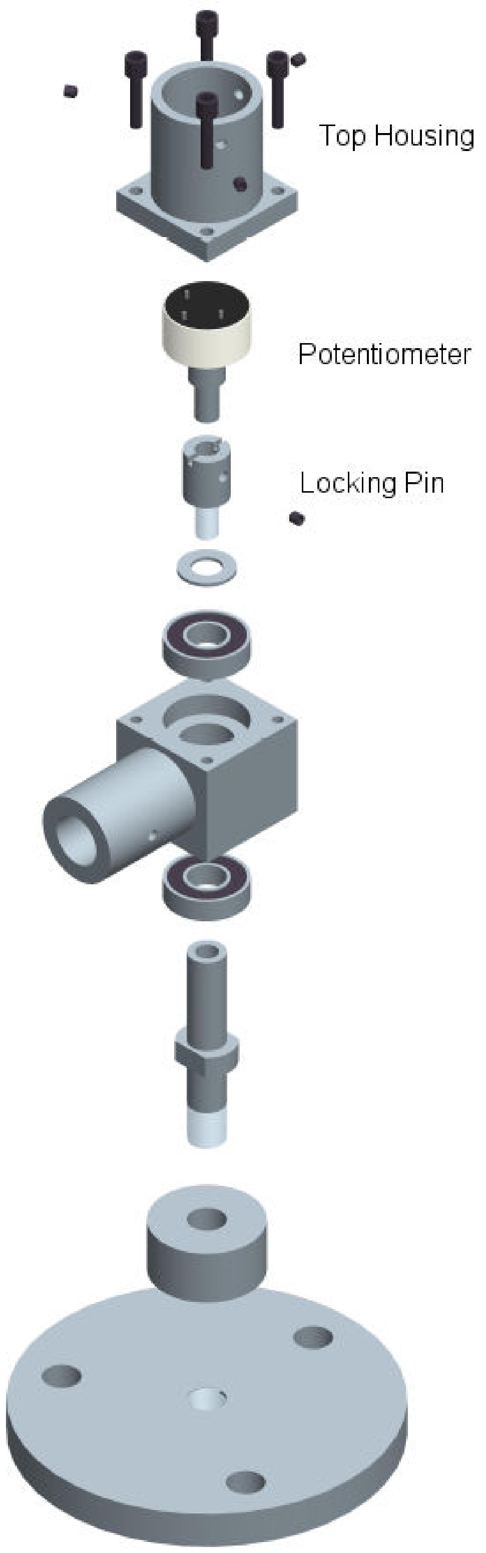


Figure 4.2: Exploded Housing

4.6 Development Board

To be able to accomplish the task specified using the system I have designed there is a step involved to relate the physical world to the computer world. This is accomplished by using the LPC-MT-2138 development board which is shown in figure 4.3.

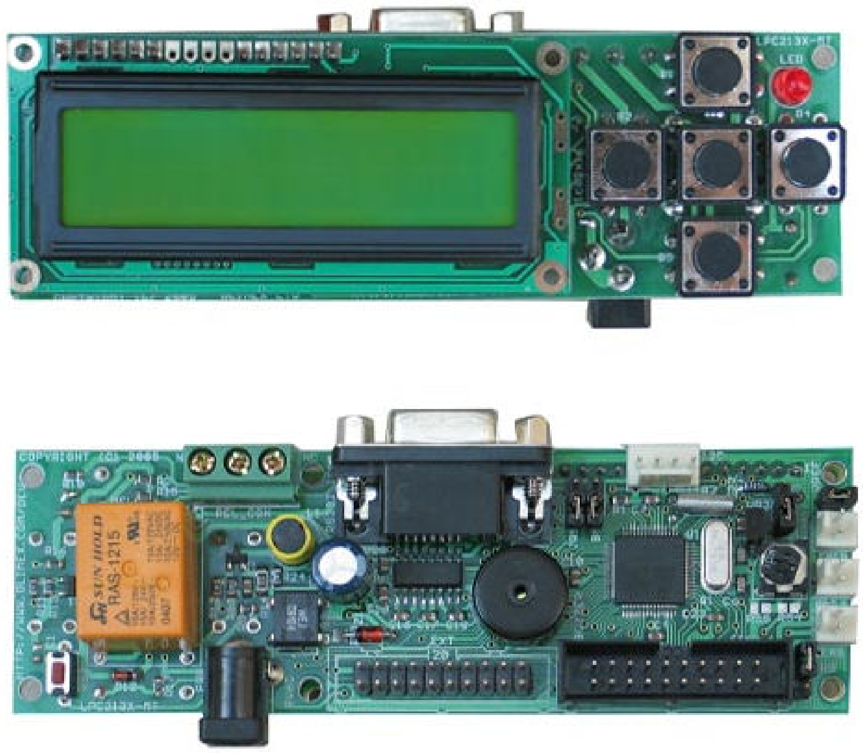


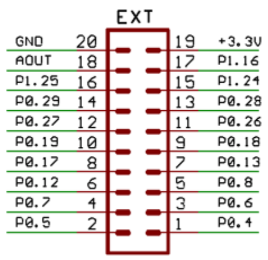
Figure 4.3: LPC-MT-2138 Development Board [25]

The relevant specifications are listed below, see Appendix E for full development board specifications.

- 58.98 MHz CPU clock
- 512 KB of programmed flash memory
- 32 KB of RAM
- 16/32 bit ARM7TDMI-S™
- 8 x 10 bit ADC
- LCD display (2 rows of 16 visible characters)
- RS232 and extension port used for interfacing

It can be seen from the specifications that the board is extremely powerful. The development board is programmed using the C language. The program used to write the code in was IAR Embedded Workbench IDE. In the following section interfacing and software development will be further discussed.

The physical connection between the development board and PC was achieved with an RS232 connection. To connection between the device and the development board was achieved by using the extension port. Figure 4.4 shows the pin arrangement for the extension port.



Pin / Name	Connected to:	Functionality
1 - P0.4	PIN27	P0.4 / SCK0 / CAP0.1 / AD0.6
2 - P0.5	PIN29	P0.5 / MISO0 / MAT0.1 / AD0.7
3 - P0.6	PIN30	P0.6 / MOSI0 / CAP0.2 / AD1.0
4 - P0.7	PIN31	P0.7 / SSEL0 / PWM2 / EINT2
5 - P0.8	PIN33	P0.8 / TXD1 / PWM4 / AD1.1
6 - P0.12	PIN38	P0.12 / DSR1 / MAT1.0 / AD1.3
7 - P0.13	PIN39	P0.13 / DTR1 / MAT1.1 / AD1.4
8 - P0.17	PIN47	P0.17 / CAP1.2 / SCK1 / MAT1.2
9 - P0.18	PIN53	P0.18 / CAP1.3 / MISO1 / MAT1.3
10 - P0.19	PIN54	P0.19 / MAT1.2 / MOSI1 / CAP1.2
11 - P0.26	PIN10	P0.26 / AD0.6
12 - P0.27	PIN11	P0.27 / AD0.0 / CAP0.1 / MAT0.1
13 - P0.28	PIN13	P0.28 / AD0.1 / CAP0.2 / MAT0.2
14 - P0.29	PIN14	P0.29 / AD0.2 / CAP0.3 / MAT0.3
15 - P1.24	PIN32	P1.24 / TRACECLK
16 - P1.25	PIN28	P1.25 / EXTIN0
17 - P1.16	PIN16	P1.16 / TRACEPKT0
18 - AOUT	PIN 9	P0.25 / AD0.4 / AOUT
19 - +3.3V	+3.3V	-
20 - GND	GROUND	-

Figure 4.4: Extension Port Pin Configuration [25]

Figure 4.5 illustrates the physical connections made between the device and development board.

AD0.6 is used on the middle potentiometer and AD0.7 is used for the potentiometer built into the fixed end. The other ADC is used for the button located on the top of the handle. The reason that an ADC is being used for input detection is related to my programming ability. I spent significant time working on using one of several different types of pins but was unable to make any progress. Since at the time the input selection for the ADC's was working appropriately I choose to use one extra ADC for the button switch detection due to time constraints.

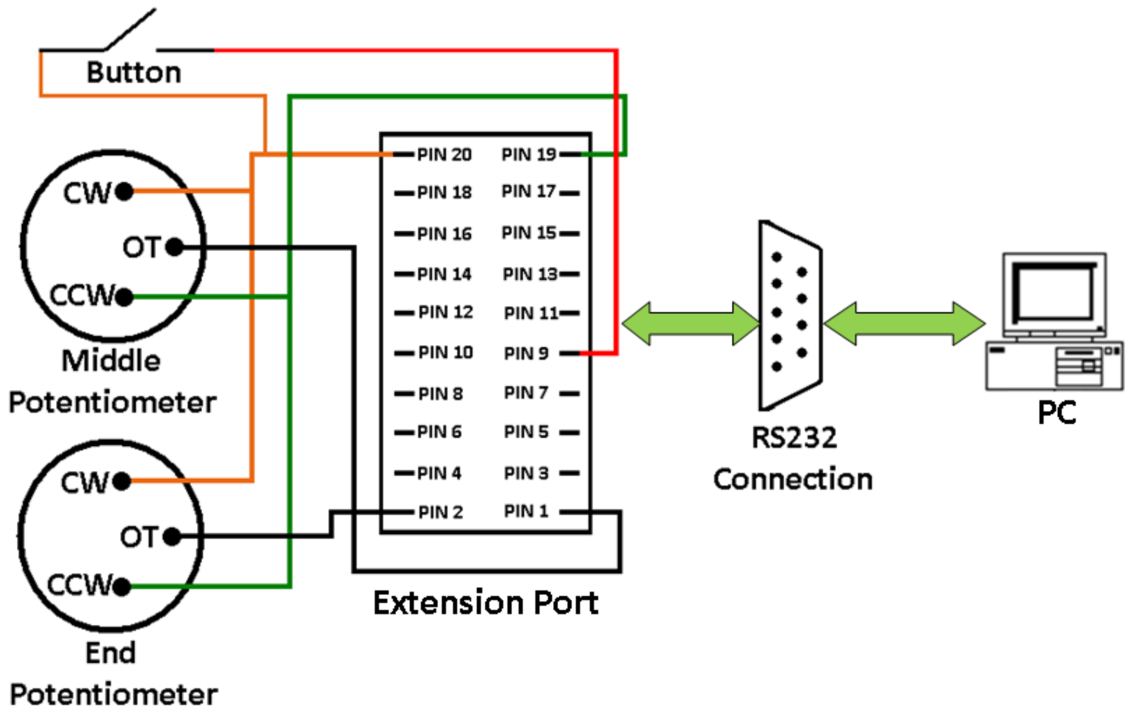


Figure 4.5: Communication Configuration

Using an ADC for button detection did lead to further problems. The problem that arose was related to noise. When the button is pushed and the voltage delivered to the ADC with virtually no resistance the ADC output would be equal to 0. As the voltage was removed noise would cause the ADC to generate random values, these would span the entire 10 bit range of 0 to 1023. The values would be constantly changing and have no apparent pattern. To overcome this problem I created a loop where every 10000 clock cycles ($\approx$ 0.01 of a second) it would read a value from the ADC and add it to the variable “checkcount” if this variable is equal to 0 after 5 cycles it is assumed that the button has been pushed rather than noise. The preferable solution would include a standard pin to be used which would call an interrupt. When the GUI program receives its first “button push” the program enters a permanent loop which completes the 3 tasks below.

- Read ADC 6/write to RS232
- Read ADC 7/write to RS232
- Check for active button, if active write to RS232

The program structure can be seen most clearly in figures 4.6 and 4.7.

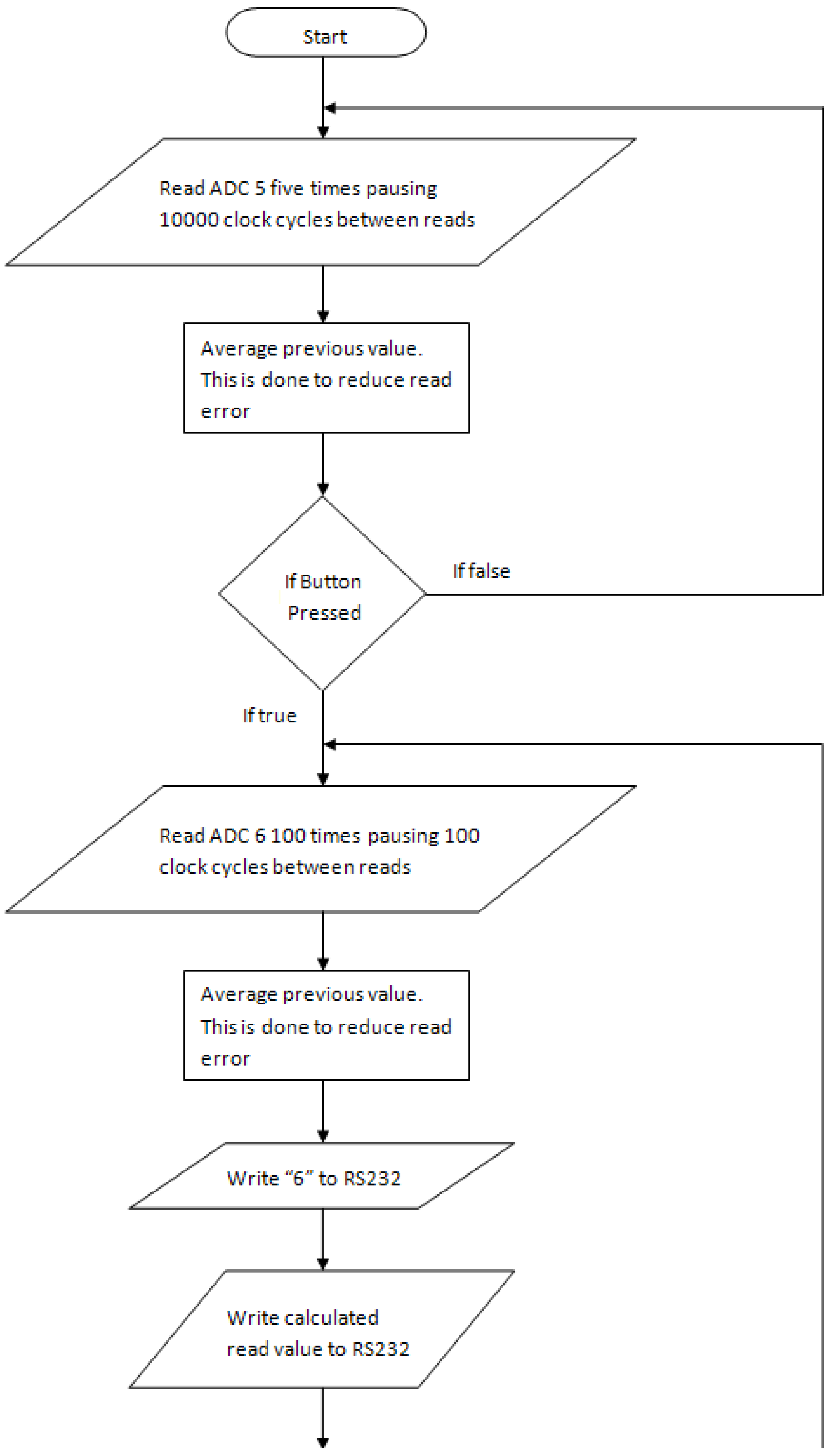

Start
Read ADC 5 five times pausing 10000 clock cycles between reads
Average previous value. This is done to reduce read error
If Button Pressed
If false
If true
Read ADC 6 100 times pausing 100 clock cycles between reads
Average previous value. This is done to reduce read error
Write "6" to RS232
Write calculated read value to RS232

Figure 4.6: Data Acquisition/Communication Flowchart

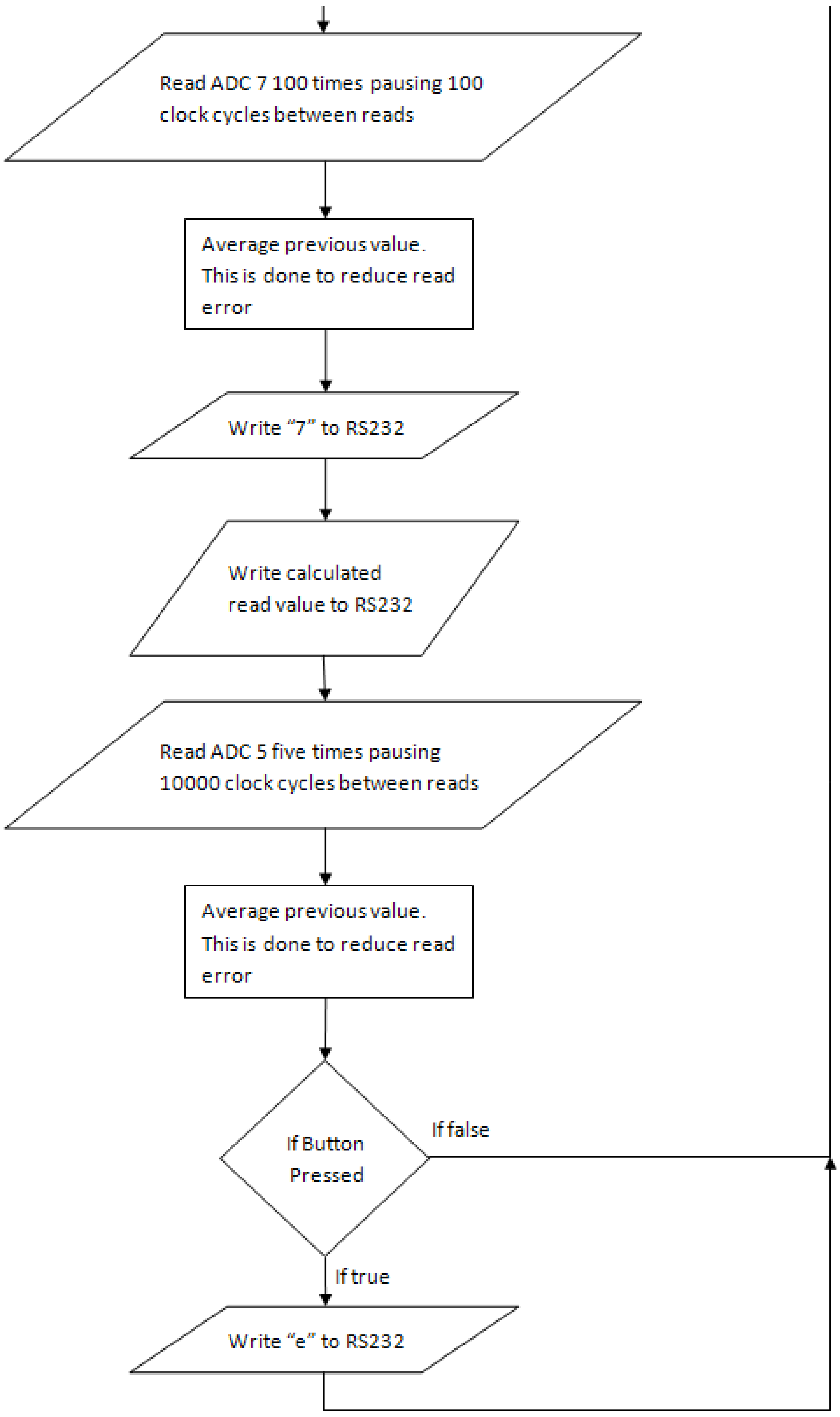

Read ADC 7 100 times pausing 100 clock cycles between reads
Average previous value. This is done to reduce read error
Write "7" to RS232
Write calculated read value to RS232
Read ADC 5 five times pausing 10000 clock cycles between reads
Average previous value. This is done to reduce read error
If Button Pressed
If false
If true
Write "e" to RS232

Figure 4.7: Data Acquisition/Communication Flowchart

The program will at first sit in a permanent loop continually reading ADC5 (the ADC designated to reading the button). This has been structured in such a manner for the user to be able to calibrate the system at the start of each session. Once the calibration has been achieved the program will enter a loop forever until the board is manually reset.

I found that there were often small read errors occurring between the potentiometers and ADC's. This seems like a small problem but resulted in the location of the cursor in my later programming to flicker backward and forwards on the screen. To attempt to overcome this read error I created a loop very similar to the button loop.

Upon entering this loop the ADC in would be called and the resulting value stored. A small delay of 100 clock cycles would then occur. It would repeat this process continually adding the values and delaying 100 times before exiting the loop. The final value is then divided by 100 to give the average value. By doing this the correct value should be occurring at a much higher rate thereby resulting in the correct answer every time. During the initial calibration mentioned earlier the ADC reading value is saved under variable names "midpot" and "endpot" to signify that these are the values which relate to the arm being located centrally in the work area. Calibration will be described in the preceding section in greater detail. The calibration values will then become part of the computations for angle change.

The values retrieved from the ADC will be subtracted from the values stored in "midpot" and "endpot". This will obviously result in positive and negative values depending on which direction the system is moved. The reason this is performed on the development board is for it to do as many computations as possible so it will remove some strain from the PC's processor and increase output. The resulting value is then sent to the PC via RS232 with a simple protocol defining which potentiometer was queried and the value calculated. This protocol can be seen in figure 4.8 and figure 4.9.

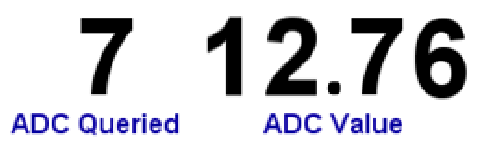


Figure 4.8: Coding Protocol

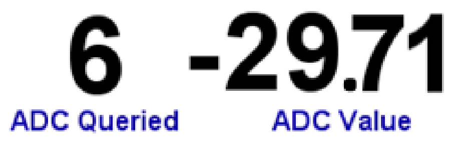


Figure 4.9: Coding Protocol

The ADC value can be either positive or negative depending on the direction that the potentiometer has rotated. This protocol is quite simple and as efficient as I could possibly achieve without losing crucial information. What occurs when the values are received at the PC is that the system will idle until calibration has been completed. The program will then load grab the correct string of values from the RS232 port and sort the values according to the starting value. The protocol will be stripped from the string of characters, converted to an integer and used within the program.

4.7 Graphical User Interface

The program that was selected to create the GUI was visual studio. Visual Studio was developed by the Microsoft Corporation to make programming a simpler task. The version of Visual Studio that was used during the development of the program was the 2005 version.

Visual Studio supports many different languages which include Visual C++, Visual C# and Visual J#, but the language which we are interested in programming in is Visual Basic. Visual basic is an all purpose language that was created to make programming object oriented [26]. The program was thus labeled basic due to this and stands for **B**eginner's all **P**urpose **S**ymbolic **I**nstruction **C**ode.

The version of visual basic being used is based around the .net (pronounced: dot net) framework. Earlier version of visual basics such as VB6 where not based around the .net framework. The .net framework can be imagined to be a layer of software built on top of the windows OS. This layer is equivalent to the windows OS being constructed on top of DOS.

The benefit of using .net software is that the programmer is able to select which language they wish to program in and the framework and its components will handle its execution.

4.8 Equipment Software

The following sections are detailed explanations of all the functions and background operations of the application which were created to work in sync with the application developed to handle and compute the ADC calculations from the LPC-MT-2138 development board.

4.8.1 The Program

On the initial startup of the program you are shown a splash screen detailing a title, software version and creator for a period of two seconds. This layout is shown below in figure 4.10. The addition of a splash screen aids in the initial visual presentation to give the software a more professional display.

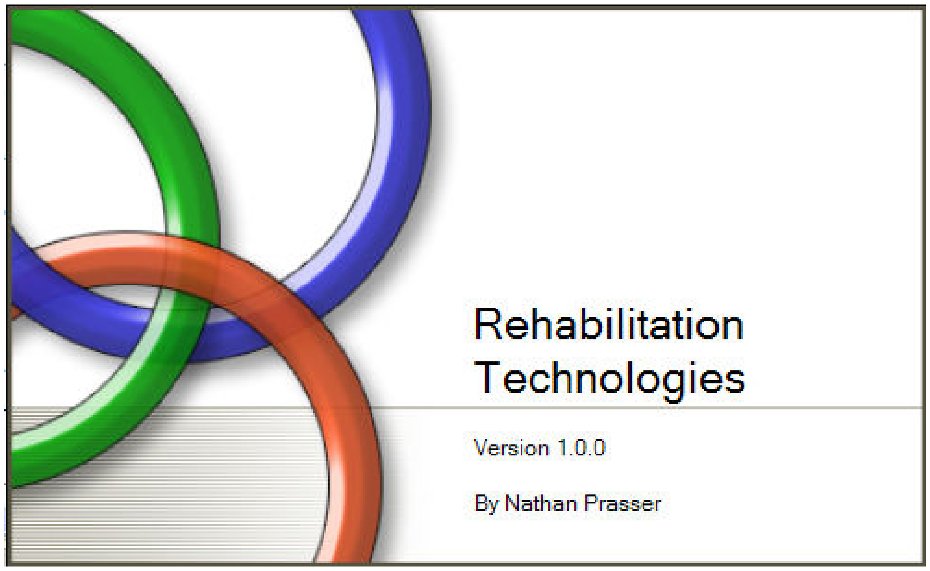
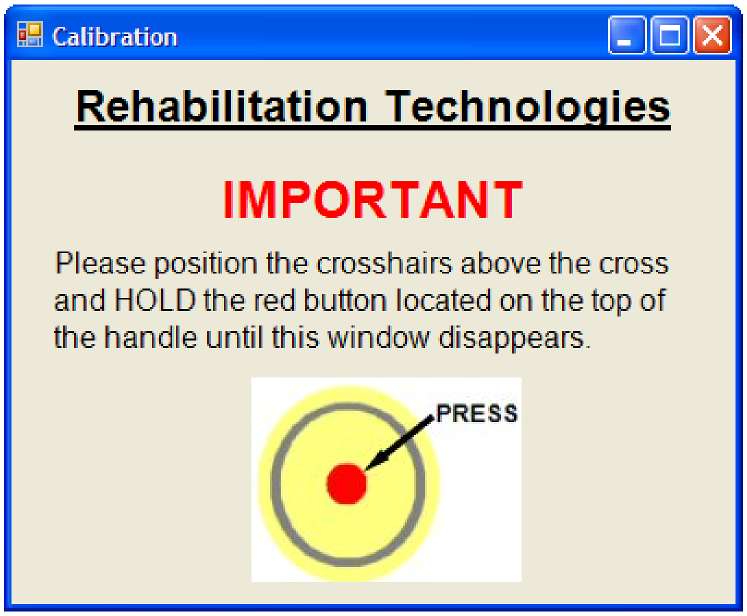


Figure 4.10: Splash Screen

After the two second period the splash screen will disappear and the calibration screen will appear. The calibration window can be seen below in figure 4.11. This step will assist in the proper sync between the GUI software and the development board software.

The use of this calibration screen allows the user to define the work area to complete their exercises for correct operation. For this to function properly the user must locate the cross hairs located on one side of the handle (the side the cross hairs are located depends on the setup for right hand/left hand operators) above the centre of the work area which is marked as cross on the workbench and to also ensure that the development board has been reset from the previous use, the screen should display the text "HOLD BUTTON". The user is then advised on the screen to hold the red button in which is located on the top of the handle. This button should be held in until the window disappears. Once the calibration has been completed the window will disappear the mouse cursor will be located centrally in the main window of the application.

After the calibration window has been completed the main application window will appear. This can be seen in figure 4.12. One of the main goals to be achieved during the development of this technology is to take the mundane task of completing physical rehabilitation and mask it with something more interesting. Therefore I have created a simple game to prove this concept.



Rehabilitation Technologies

IMPORTANT

Please position the crosshairs above the cross and HOLD the red button located on the top of the handle until this window disappears.

Figure 4.11: Calibration Screen

For the simplicity of programming I have used the values obtained from the development board to control of the mouse cursor. The reasons for doing so will become apparent later in this section. The game consists of moving the handle of the device in a coordinated with a target onscreen.

Once the user has reached the green target they must stop on its location, when this has been accomplished the green target will move to a new location and require the user to once again move to this new location. If the user moves continuously on the target or passes over the top without stopping on it nothing will happen. An example of the location change can be seen in figure 4.13.

(Discussion continued on page 60)

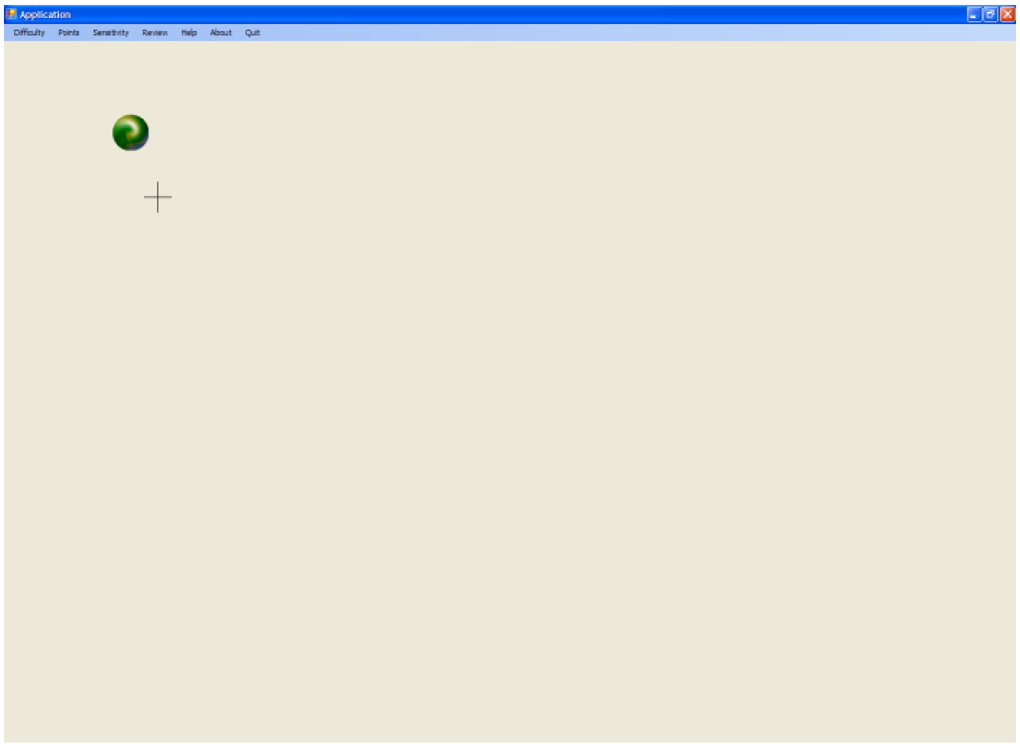


Figure 4.12: Approaching Target

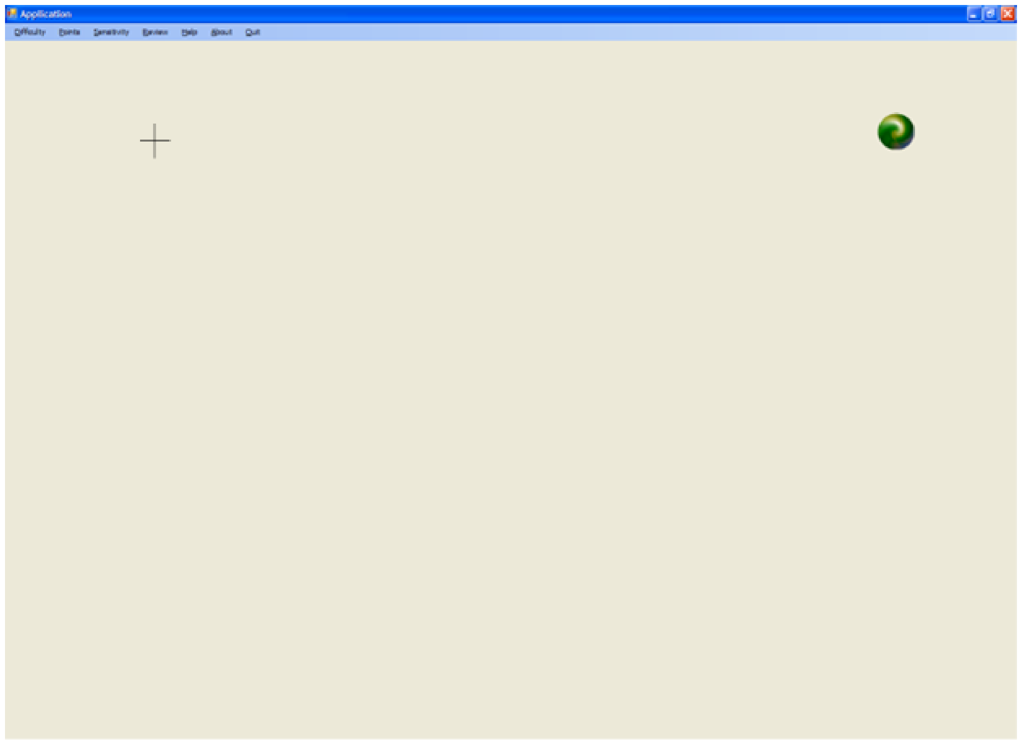


Figure 4.13: Target Location Change

The button located on top of the handle that was used originally for the calibration now has a different function within the program. As we know the mouse cursor is being controlled by the values determined from the device relative to its location. But to use other functions within the program we are going to need to be able to regain control of the mouse. To do this we use the red button. By pressing this button we are able to interchange between the device for control and the mouse for control. The mouse will allow different functions of the application to be explored.

This is very important for a fully operational and functional program. To assist the viewer in seeing when the device is active a small message will be displayed in the top right hand corner when the device is active. An example of this can be seen in figure 4.14. If the user attempts to operate the mouse while the device is active the mouse cursor will move only a small distance before it is returned to its location according to the values being calculated from the development board.

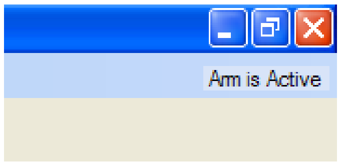


Figure 4.14: Active Arm Notification

Figure 4.15 demonstrates what occurs when the mouse is used while the device is activated.

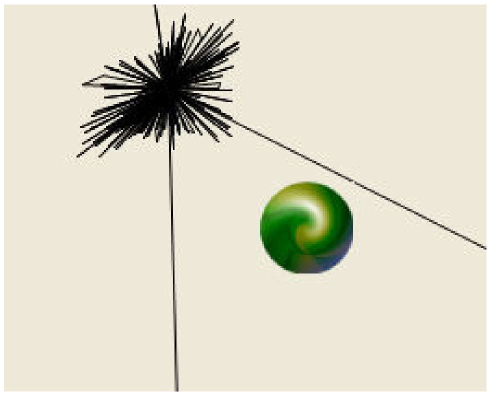


Figure 4.15: Mouse Disturbance

4.8.2 Therapy Options

During the development of the GUI I created as many options possible to introduce flexibility into the system for users with different levels of competency. Figure 4.16 and figure 4.17 show an option of exercising with one of three possible difficulties each relating to a different size target. As the difficulty is increased the circular target reduces in size and also becomes located further away from the center of the screen. This allows an extra level of difficulty to be added for users who are progressing with muscle control and coordination and require a more challenging game.

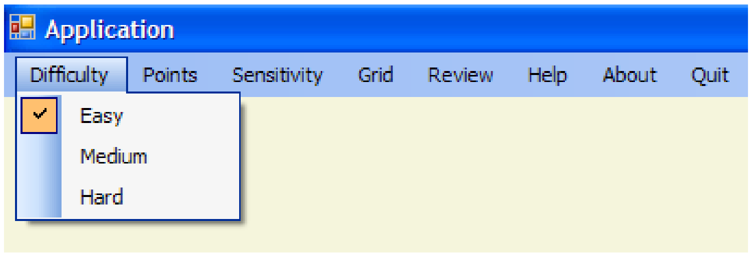


Figure 4.16: Difficulty Options

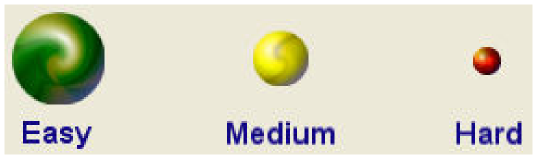


Figure 4.17: Targets Available

There are two different options relating to the target locations. The first option is to arrange the targets into a known set of locations referred to as “Test Locations”. These locations will always fall into the general shape of a triangle. The location of these targets will automatically change upon the application window being manually resized to remain in the required triangular pattern. It will become more apparent why this function is available later in this section.

The second option is to allow the location of the target to be placed in a completely random position anywhere on the screen. This will allow users to overcome the repetitiveness associated with the Test Locations. This function is referred to as “Random Locations”. These options can be seen in figure 4.18.

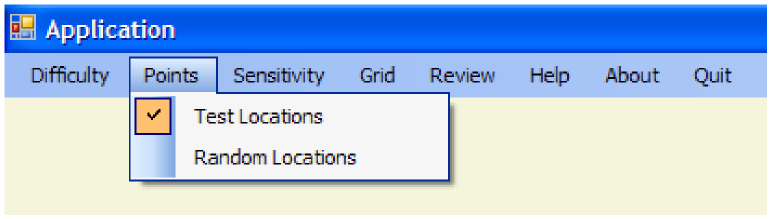


Figure 4.18: Target Locations

During the operation of the device you will be able to adjust the sensitivity of the movements made relative to on screen movements. This once again allows the ability of the user to select a setting which most suits their needs. This can be seen in figure 4.19.

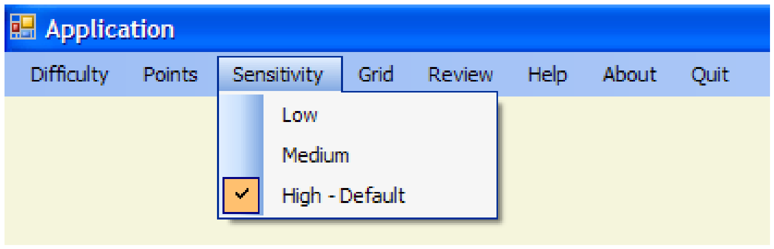


Figure 4.19: Device Sensitivity

In the figures below, figure 4.20 and 4.21 there is an option to enable a graph pattern to be laid over the work area to assist in directing movement and will also be used in the following chapter for the analysis of results.

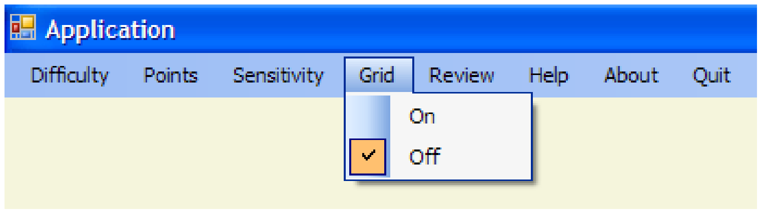


Figure 4.20: Grid Toggle

One of the greatest things about the technology under development is the ability to track all movements made by the user. This allows the patient and physiotherapist to accurately monitor the progress of the patient. The ability to log movements has only been enabled for the “Test Locations” option mentioned above. If tracking where to be used for “Random Locations” the person reviewing the progress would not be able to understand whether the movements made where accurate or logical. For the simplicity of interpreting the results the “Test Locations” are limited to three locations per difficulty. Below in figure 4.22 is an example of reviewing the logged movements for session with the difficulty set as “easy”.

Controlling the mouse cursor with the calculated output derived from the development board has allowed me to save significant time being the mouse cursor has many useful subroutines predefined available for it. MouseHover was a very important subroutine used within my program. As the mouse cursor is placed over the green shape for instance, I would be able to call for one image to be hidden from view and another to be shown to simulate a singular moving image.

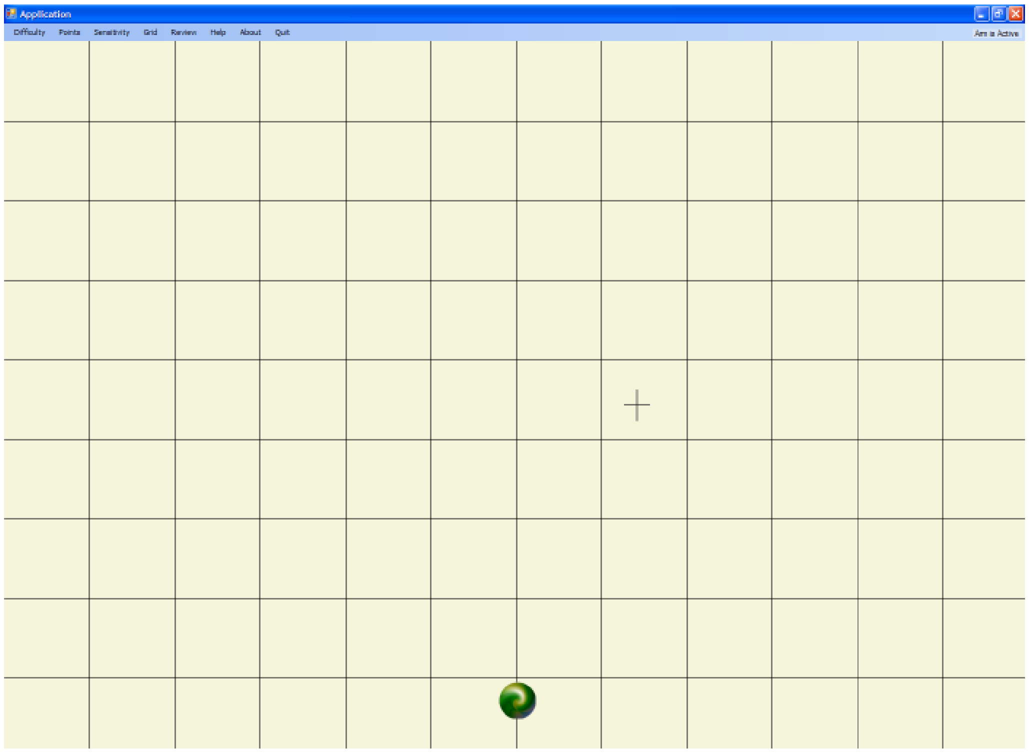


Figure 4.21: Grid Pattern

The most useful functions that I used from the inbuilt mouse subroutines were the paint and MouseMove subroutines. The two subroutines were used in coordination with one another. During the session where the user completes the game, the MouseMove subroutine stores all the location points which have been covered by the mouse cursor. When the review form is called the paint function can simply read these location points stored previously and recreate the movements.

The cursor location values are held in two list boxes, when a new session is created these values are exported to a unique location and stored within two files text files. The benefits of storing the information as a list of values in text files are that the files remain small and the information if required is easily manipulated and imported into other programs such as Microsoft excel. One file containing the information relating to the x coordinates and the other relating to the y coordinates. The axes are stored as separate files to make writing and reading a simpler task. These files will be automatically be created within the folder the application is stored.

Figure 4.23 and figure 4.24 shows how the review section of the application is arranged. The reviewer will first specify from which date they wish to review from the drop down box. This will display the full details of the location of the file. After the date has been selected the second drop down box will be enabled and a specific session will be able to be reviewed. Once both have been selected the reviewer simply presses the “GO” button located besides the drop down boxes and the session will be visible.

Figure 4.22 is a good example of what will be visible while reviewing past sessions. It is also helpful to see when selecting from the past sessions in the second dropdown box the difficultly which was undertaken during the session is displayed at the trailing end of the file location.

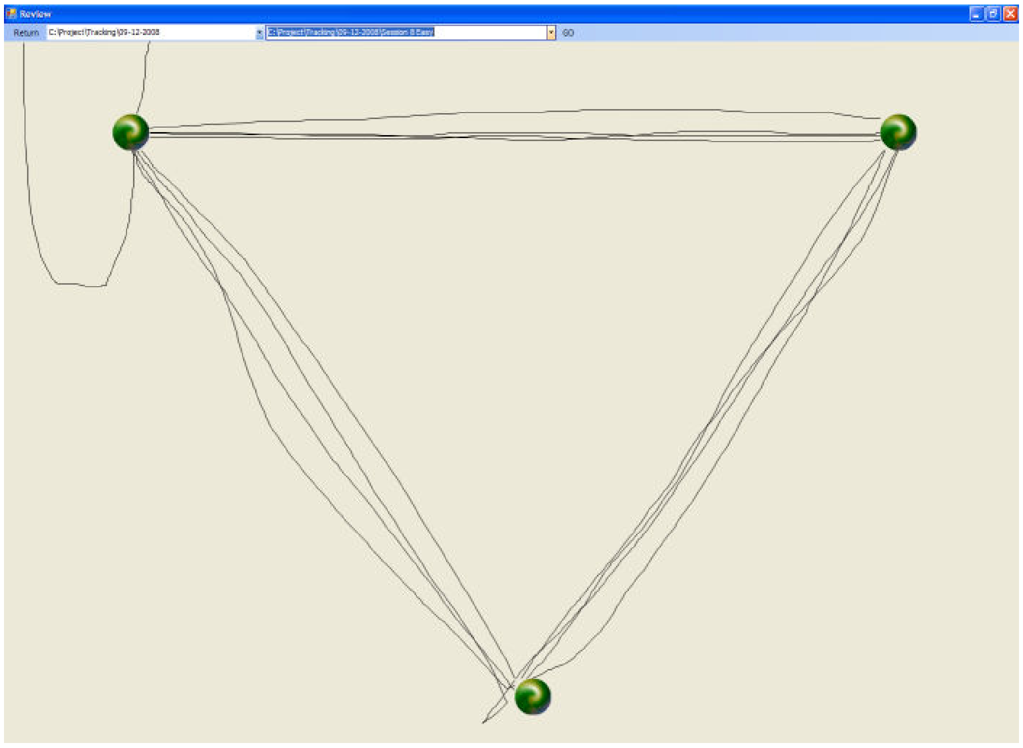
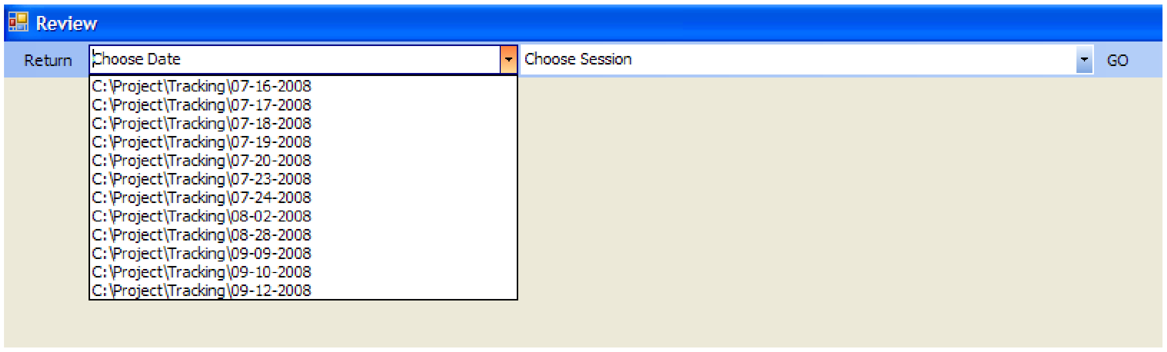
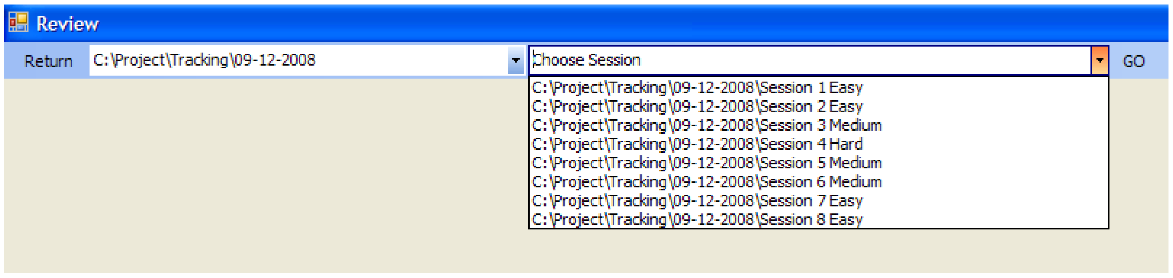

Review
Return
C:\Project\Tracking\09-12-2008
C:\Project\Tracking\09-12-2008\Session 8 Easy
GO

Figure 4.22: Reviewing Movements



C:\Project\Tracking\07-16-2008
C:\Project\Tracking\07-17-2008
C:\Project\Tracking\07-18-2008
C:\Project\Tracking\07-19-2008
C:\Project\Tracking\07-20-2008
C:\Project\Tracking\07-23-2008
C:\Project\Tracking\07-24-2008
C:\Project\Tracking\08-02-2008
C:\Project\Tracking\08-28-2008
C:\Project\Tracking\09-09-2008
C:\Project\Tracking\09-10-2008
C:\Project\Tracking\09-12-2008

Figure 4.23: Date Option



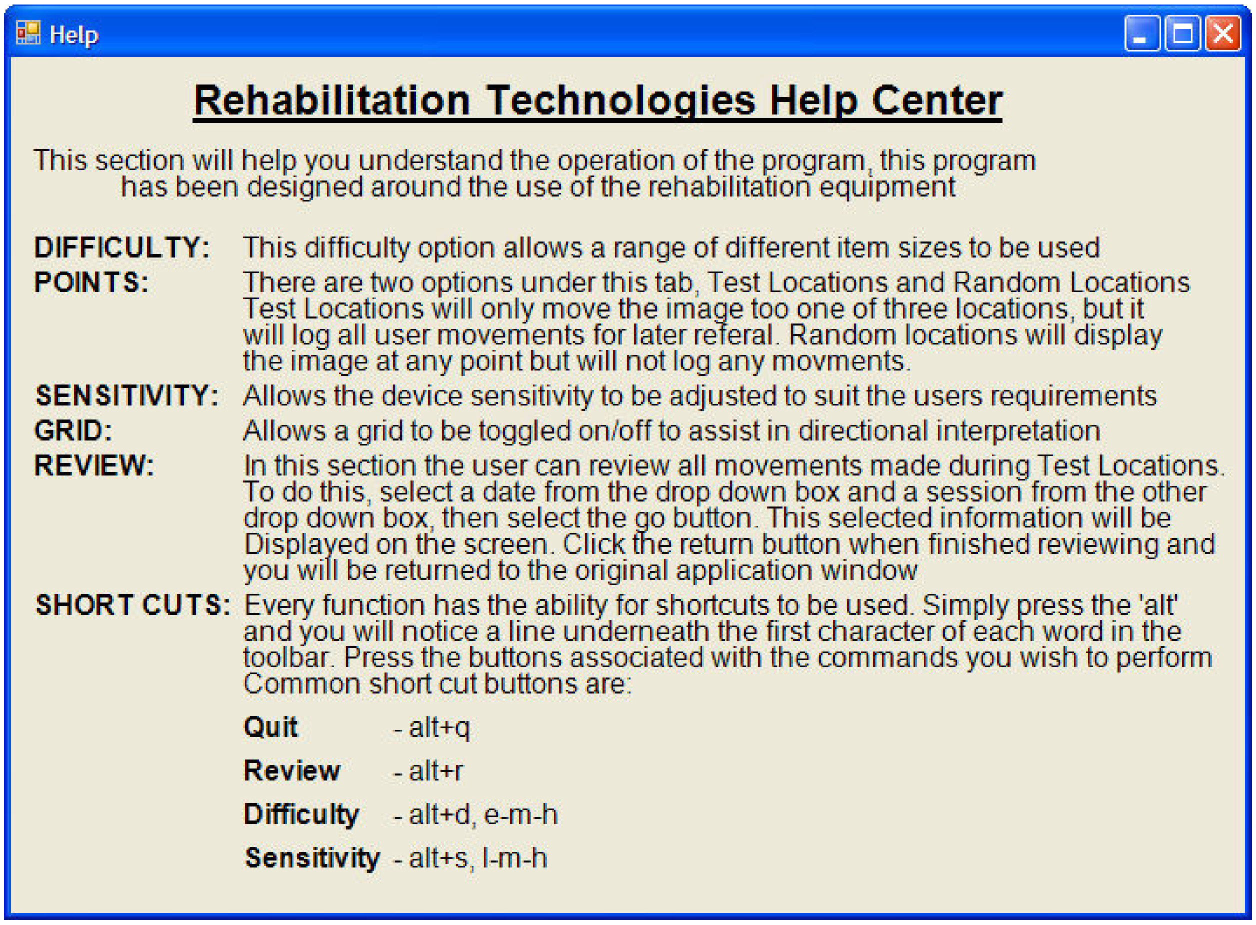
C:\Project\Tracking\09-12-2008

Choose Session

C:\Project\Tracking\09-12-2008\Session 1 Easy
C:\Project\Tracking\09-12-2008\Session 2 Easy
C:\Project\Tracking\09-12-2008\Session 3 Medium
C:\Project\Tracking\09-12-2008\Session 4 Hard
C:\Project\Tracking\09-12-2008\Session 5 Medium
C:\Project\Tracking\09-12-2008\Session 6 Medium
C:\Project\Tracking\09-12-2008\Session 7 Easy
C:\Project\Tracking\09-12-2008\Session 8 Easy

Figure 4.24: Session Option

The application window has a small inbuilt help center to explain the usage and function of all available tools throughout the program. This help center is displayed in figure 4.25.



Rehabilitation Technologies Help Center

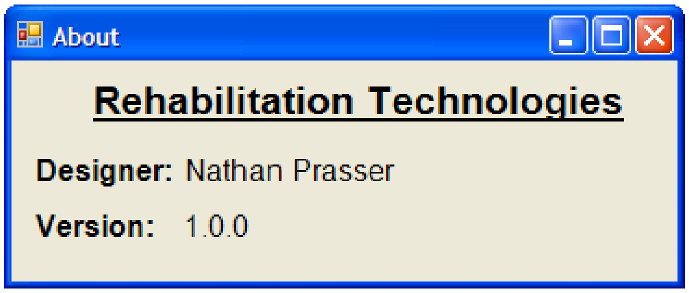
This section will help you understand the operation of the program, this program has been designed around the use of the rehabilitation equipment

DIFFICULTY:	This difficulty option allows a range of different item sizes to be used
POINTS:	There are two options under this tab, Test Locations and Random Locations Test Locations will only move the image too one of three locations, but it will log all user movements for later referal. Random locations will display the image at any point but will not log any movments.
SENSITIVITY:	Allows the device sensitivity to be adjusted to suit the users requirements
GRID:	Allows a grid to be toggled on/off to assist in directional interpretation
REVIEW:	In this section the user can review all movements made during Test Locations. To do this, select a date from the drop down box and a session from the other drop down box, then select the go button. This selected information will be Displayed on the screen. Click the return button when finished reviewing and you will be returned to the original application window
SHORT CUTS:	Every function has the ability for shortcuts to be used. Simply press the 'alt' and you will notice a line underneath the first character of each word in the toolbar. Press the buttons associated with the commands you wish to perform Common short cut buttons are:

Quit	- alt+q
Review	- alt+r
Difficulty	- alt+d, e-m-h
Sensitivity	- alt+s, l-m-h

Figure 4.25: Help Center

For ease of reference the main window has an “about” tab which quickly shows important information such as the version of the program. An example of this window is shown in figure 4.26.



Rehabilitation Technologies

Designer: Nathan Prasser

Version: 1.0.0

Figure 4.26: About Window

4.8.3 Communication Form

The operation which reads the serial port has been labeled RS232COM. The operation of this form relies on several textboxes to handle and strip the communication protocols. The full listing of these textboxes is displayed in figure 4.27 and 4.28.

The development board has been setup to constantly stream new values to the serial port. There is no communication between the application and the development board to call for a value and then have it returned. I have used a unidirectional communication method to increase the speed of transfer and to also simplify the programming as much as possible.

The protocol I have defined for the data transfer is as follow:

- 8 data bits
- 1 stop bit
- 0 parity bits
- Communication to always occur on COM port 1
- Baud rate 9600bps

The RS232COM form first invokes the serial port then it saves this value into the location referred to as “current RS232 read value” in figure 4.27 and 4.28. Once a value has been received it is transferred to “copy of current RS232 read value” where it can be used later. The program will read the communications channel for a second time. The value stored in the textbox “copy of current RS232 read value” will be transferred to “copy of previous RS232 read value”. The value which was read the second time will then be transfer to “copy of current

RS232 read value”. If the value stored in “copy of previous RS232 read value” contains a “6” or a “7” it will transfer the value stored in “copy of current RS232 read value” to the either “most recent middle potentiometer value” or “most recent end potentiometer value”. The field is defined by where the previous value obtained was a “6” or a “7”, “6” would be transferred to the middle potentiometer textbox were as a “7” would be transferred to the end potentiometer textbox. A flow diagram of this code can be seen in figure 4.27. An example of the transfer is best seen in figures 4.28 and 4.29.

Initially if the parameters were not correct it would continue to read and store in the “current RS232 read value” until the system would crash. Therefore to overcome this I added another textbox called “character count” that would count the total number of characters in the string and if this exceeded a certain value the textbox would be cleared and a read would be attempted again.

Once a successful read has taken place and has been stored in the appropriate textboxes, a different section of the program will call these values for handle location calculations. These textboxes are both being updated very quickly to enable a smooth transition of values.

(Discussion continued on page 69)

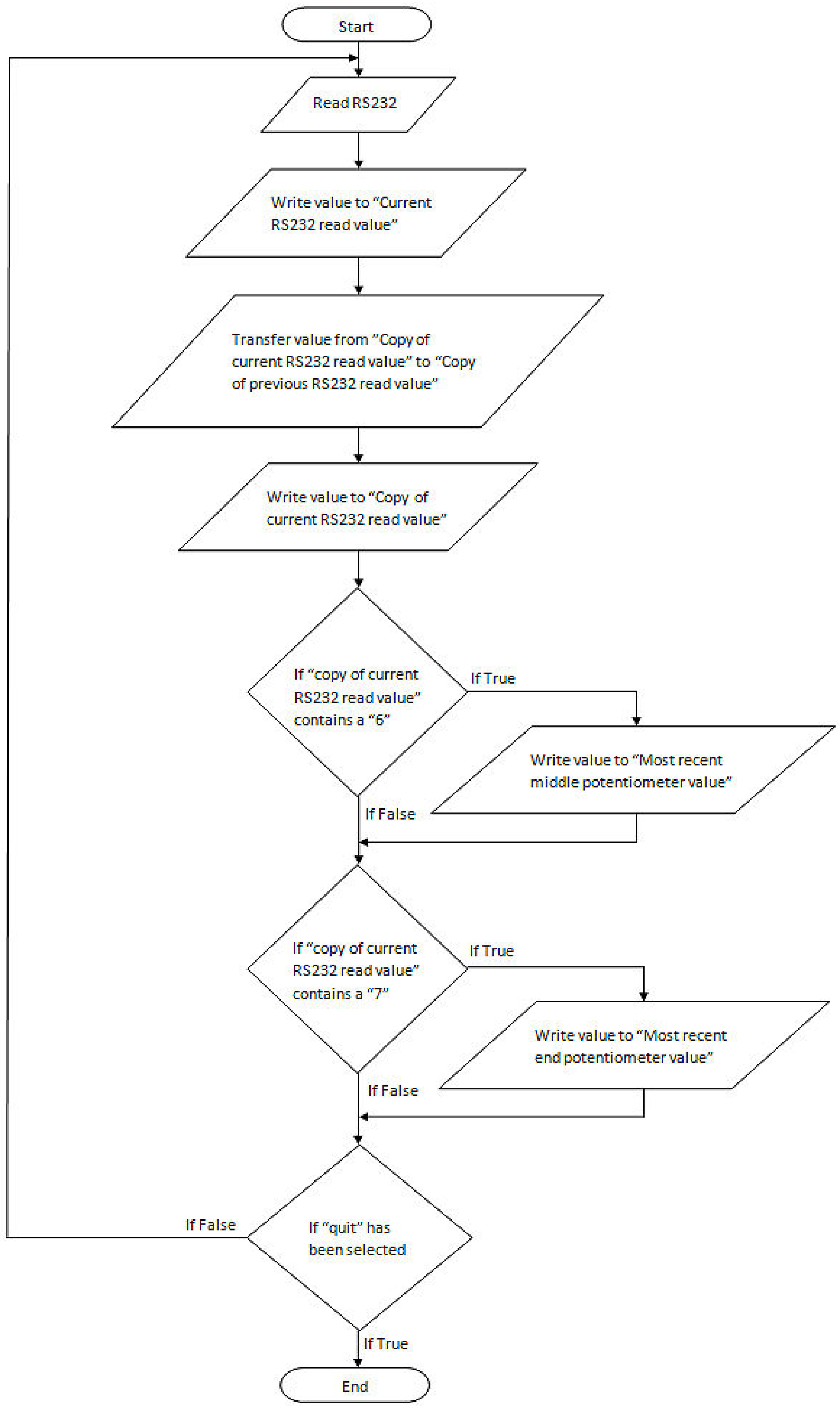

Start
Read RS232
Write value to "Current RS232 read value"
Transfer value from "Copy of current RS232 read value" to "Copy of previous RS232 read value"
Write value to "Copy of current RS232 read value"
If "copy of current RS232 read value" contains a "6"
If True
Write value to "Most recent middle potentiometer value"
If False
If "copy of current RS232 read value" contains a "7"
If True
Write value to "Most recent end potentiometer value"
If False
If False
If "quit" has been selected
If True
End

Figure 4.27: RS232 Operation

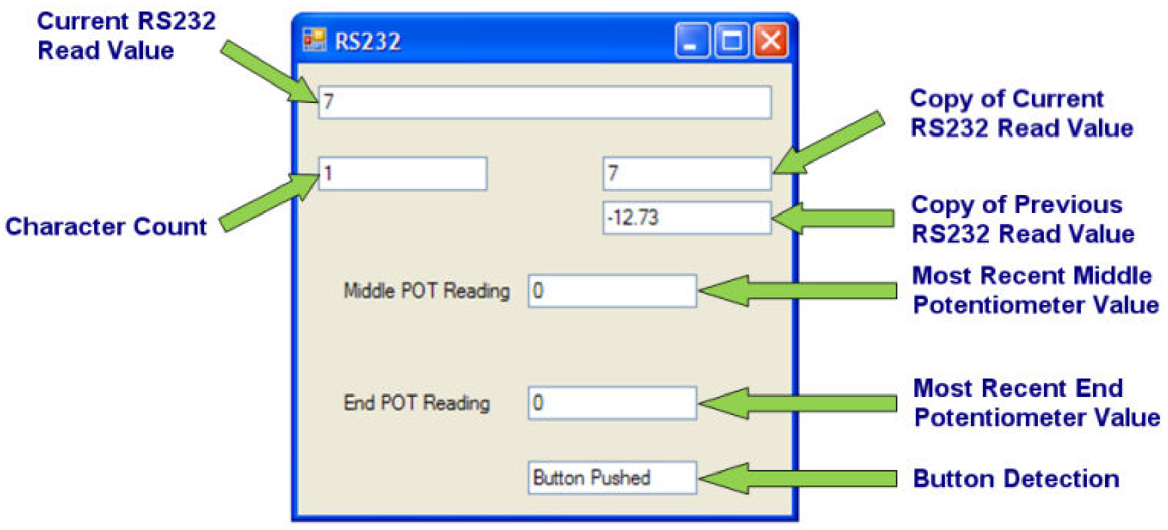

Current RS232 Read Value
Character Count
RS232
7
1
7
-12.73
Middle POT Reading
0
End POT Reading
0
Button Pushed
Copy of Current RS232 Read Value
Copy of Previous RS232 Read Value
Most Recent Middle Potentiometer Value
Most Recent End Potentiometer Value
Button Detection

Figure 4.28: Communication Form (first read)

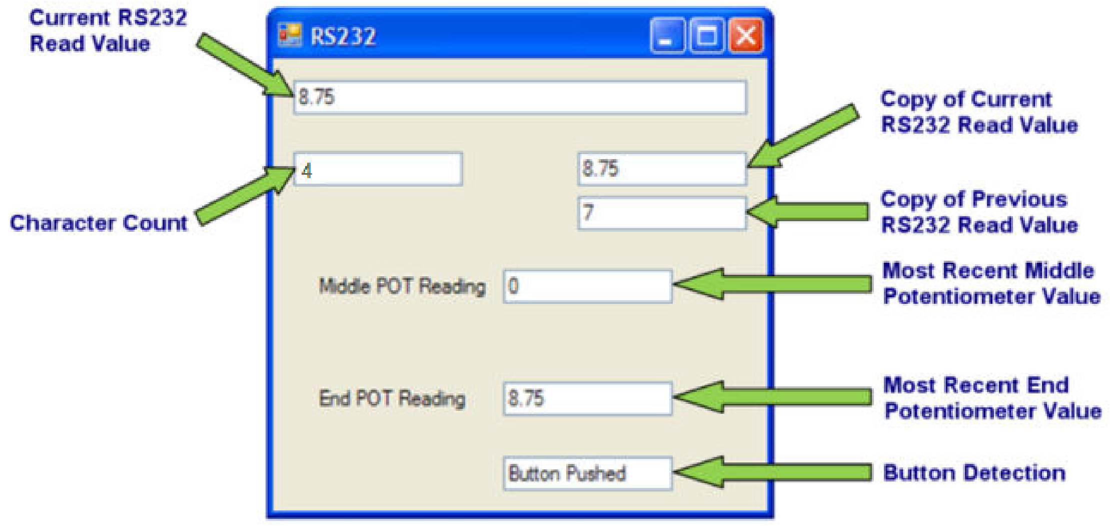

Current RS232 Read Value
Character Count
RS232
8.75
4
8.75
7
Middle POT Reading
0
End POT Reading
8.75
Button Pushed
Copy of Current RS232 Read Value
Copy of Previous RS232 Read Value
Most Recent Middle Potentiometer Value
Most Recent End Potentiometer Value
Button Detection

Figure 4.29: Communication Form (second read)

The values which are derived from the serial port have had some calculations previously performed on them to assist in the output speed of the final program. The following section details how the received values are manipulated as to calculate the position of the devices handle relative to the fixed end point.

4.8.4 Position Algorithms

The advantage of using a calibration process is that the handle can be located at any point and still be able to refer to that as its origin of reference. This is shown in figure 4.30. Since the system has been designed this way, whenever the joints of the device begin to rotate the resulting value will be either positive or negative depending on the direction that the rotation has occurred.

The value which is received at the serial port is the difference in resistance broken into a range from 0 to 1023 and restricted to having a float value of only two decimal places. It is not an angle of change at this point.

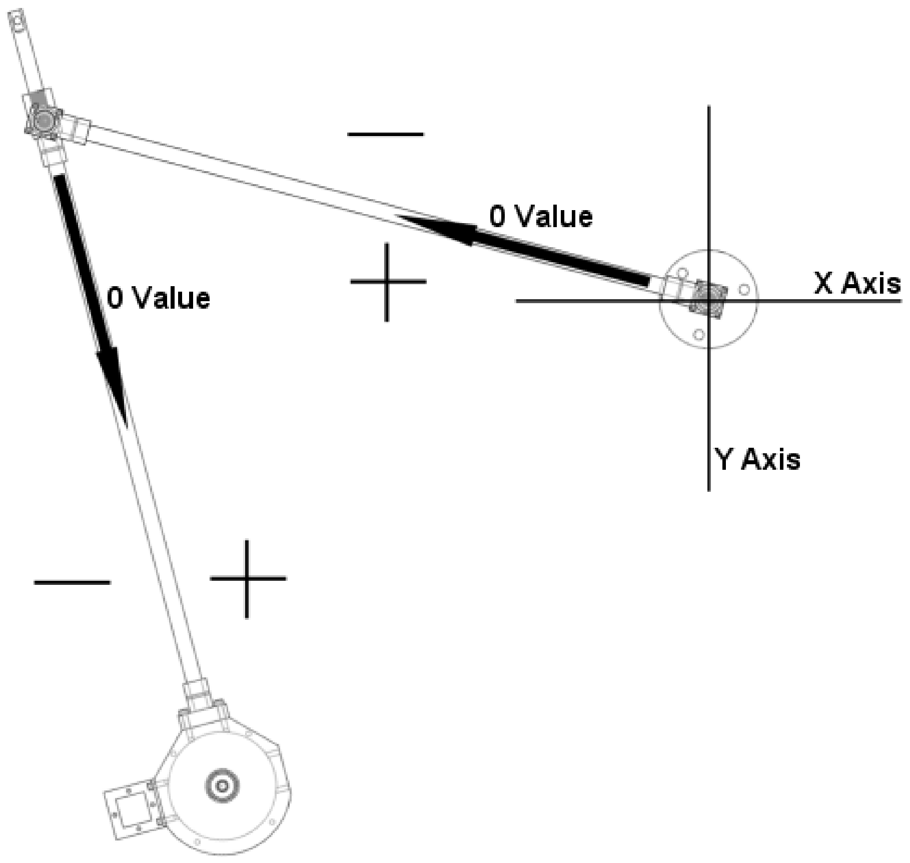


Figure 4.30: Algorithm

For the read values to be signed (+ or -) as we demand we must reverse the sign of the end potentiometer value and also modify the value that currently is “change of resistance” to “change of degrees” before it can be inserted into the algorithm shown in code 4.1. This code performs some simple trigonometry to calculated the handle of the device relative to the fixed end point. The completed code for both the GUI and development board can be seen in appendix C.

Code 4.1: Position Algorithm

```
'Calculate the x position and y position of the handle relative to the fixed
‘point
xposition = (armlength * Cos(degreechange1)) + _
armlength * Sin(degreechange1 + degreechange2))

yposition = (armlength * Sin(degreechange1)) + _
(armlength * Cos(degreechange1 + degreechange2))
```

4.9 Conclusion

This chapter has covered significant content relating to the designs developed to create a technology for stroke rehabilitation. It has covered different technologies that could be added to the system to allow active movements. A final design of the proposed system was also chosen. The final design relied heavily on the overall safety of the device. A device which had the possibility to cause harm would defeat the purpose of the technology and also lead to legal implications.

Software has been a significant topic in the chapter because without the use of software this system would represent a technology similar to those already currently being used. All the advantageous possibilities that this device is able to achieve is directly related to the quality of the software. The game created for this project is not high quality or able to hold ones attention for any length of time, but this is rather created simply to show the possibilities are there for future work in this field.

This chapter described some crucial sections relating to the software structure and operation for the program to communicate with the device and function properly. The following chapter will discuss the final device and results obtained from the final version of the software.

Chapter 5
Device Evaluation & Analysis of Results

5.1 Introduction

The following chapter discusses the work completed on the device and software. This covers testing the device and discusses the overall results of the system. The process of testing involved the physical movements of the device being compared to the movements occurring onscreen.

The method of obtaining a final result was by testing accuracy and precision of the device between repeated uses, and by the overall range-of-motion that was attainable. This section will also discuss problems that occurred during development of the program and methods used to rectify these problems if they were able to be corrected.

5.2 Method for Testing

To be able to test the three characteristics required (accuracy, precision and range-of-motion) for a final evaluation of the device, I have devised an experiment which allows the physical movements to be compared against movements occurring on the computer. This involves:

1. Positioning the cursor while being controlled by the device in all four corners of the application window. From this we are able to plot the operational work area.

2. Once the boarders have been established we are able to create a grid pattern that mimics that shown previously in figure 4.21. This grid pattern requires the work area to be broken into even sections with nine blocks vertically and twelve blocks horizontally.

3. At this stage we have recreated a physical representation of the pattern from the computer program. From this we are now able to place all the targets on the grid pattern relative to their position in the computer program.

4. We are now able to perform a physical test to mark the location of the targets according to the device results. From these new set of targets we are able to draw straight lines between each location to assist in straight movements. The completed testing board can be seen in figure 5.1.

5. Once the targets physical locations have been established we are able to start testing. To test the device it is a matter of following the straight line established in the previous

step for each level of difficulty. To allow accuracy and precision to be easily established I performed the motions three times for every single test performed. To ensure the results were accurate the software was restarted, the development board manually reset and calibration completed for each set of results. All results performed during testing are shown in appendix F.

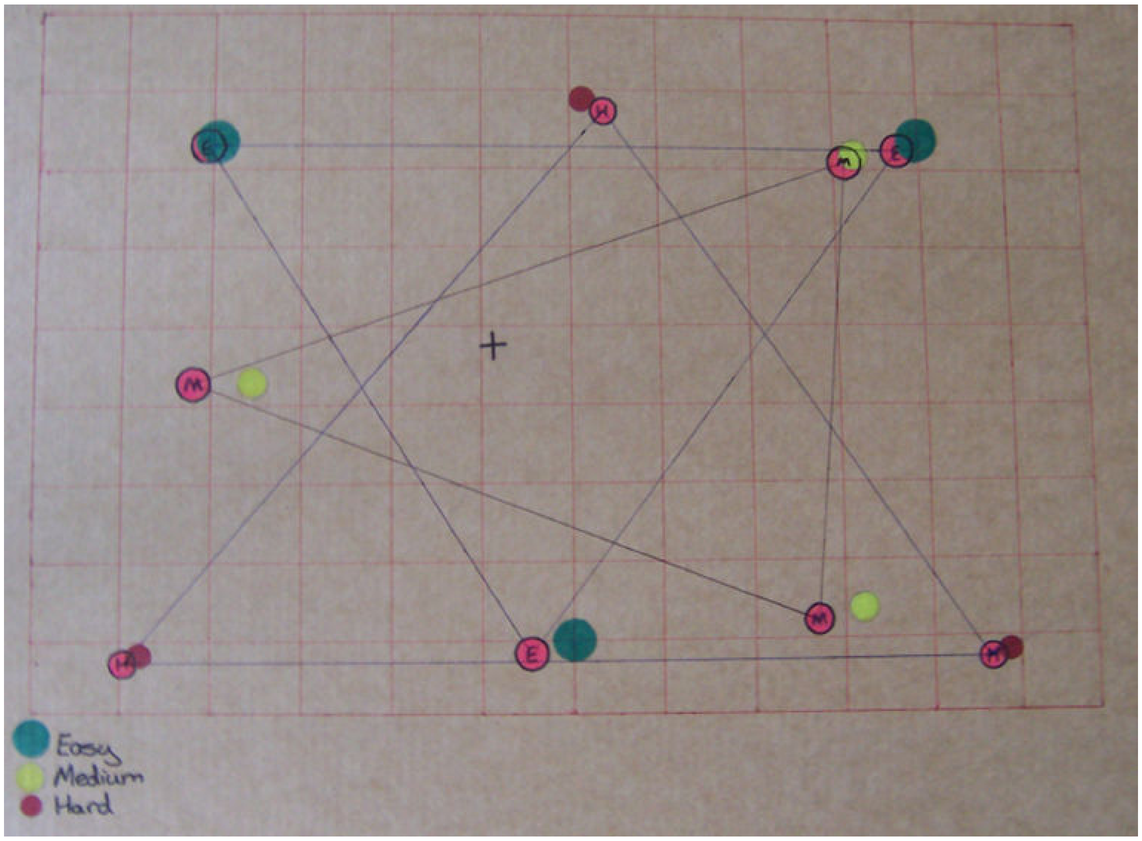


Figure 5.1: Testing Board (40 pixel offset to the left)

5.3 Range of Motion

During the development of the GUI I incorporated a function in which the user is able to adjust the sensitivity of the device. This option allows functionality for users with different needs. The work areas of the three sensitivity options are as follows:

- Low sensitivity – 570 millimetres by 845 millimetres
- Medium sensitivity – 375 millimetres by 580 millimetres
- High sensitivity – 97 millimetres by 145 millimetres

From these resulting values it can be seen that the initial maximum work area has been achieved.

5.4 Device Accuracy

The definition of accuracy is the quality of being near to the true value [27]. Accuracy is best described by the illustration shown in figure 5.2. Therefore for the device to have a high degree of accuracy we require that the physical targets locations (established in section 5.3) can be repeatedly correlated to the target locations of the onscreen actions.

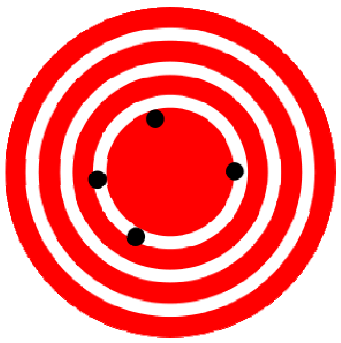


Figure 5.2: Accuracy [28]

The results obtained from the performed tests all reassembled a similar outcome. Therefore one test result will be used for the review of the accuracy and precision on the device. All performed test results are available in appendix F.

The test used for the analysis of accuracy can be seen in figure 5.3. This figure was obtained by using the review section built into the GUI program. For the analysis of results the image was modified to include target locations obtained from the physical test (red circle with a black edge). During testing, the device was moved between these targets on the test board guided along the red interconnecting line. The resulting outcome of the onscreen performance can be seen as the black lines which are linked between GUI targets (in this case a green target).

The results across the entire set of tests have indicated that the device lacks accuracy. In the majority of cases the target location in the physical exercise was located to the left hand side of the computer target. I tried to correct for the majority of displaced target locations by offsetting the cursor location through a range of various test values. I found that altering the target location values with an offset often introduced further problems into the system.

The accuracy of this device requires further investigation before additional work on the other components (introduction of an active system) of the system would be able to continue.

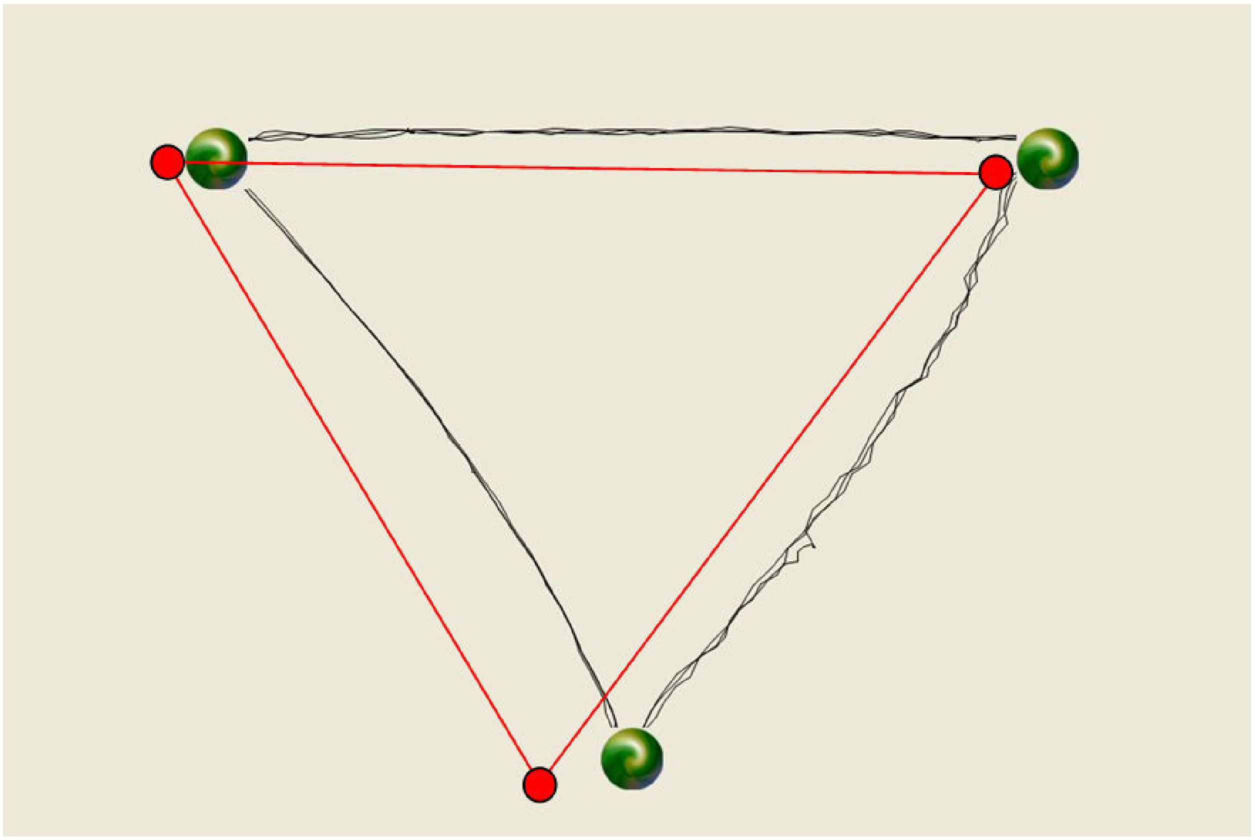


Figure 5.3: Test Result

5.5 Device Precision

The definition of precision is the quality of being reproducible in amount or performance [29]. Precision is best described as a cluster of points that are closely grouped at any location, this is seen in the illustration shown in figure 5.4. Therefore for the device to have a high degree of precision it must be able to repeatedly calculate the same location.

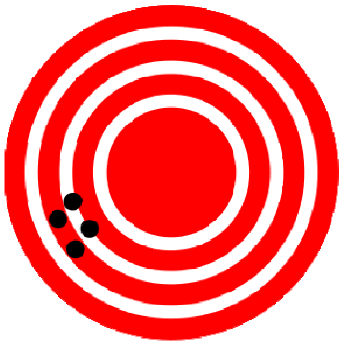


Figure 5.4: Precision [30]

Once again looking at the results obtained from the tests performed (figure 5.3 and appendix F) we are able to establish that the device has a very high rate of precision. In many cases this was determined by how precisely I was able to move the device along the exact same line repeatedly.

The figure that was chosen for determining the precision (and accuracy) was based around the abnormalities occurring along the right hand side diagonal line. This line shows less precision then the other two lines in the figure. What is occurring along this line to cause these abnormalities is read errors by the ADC's. There are two methods to make these errors less prevalent. These are:

- Dedicate more time for the ADC's to read the potentiometer values.
- Read the ADC's more times than currently being read (discussed in section 4.6) and average the values.

5.6 Analysis of Results

These three tests have allowed the performance of the device in its current state to be analyzed. The first test was to analyze if the maximum range of motion that the device operated in correlated to the required work area established in section 3.2. In section 3.2 the maximum work area from a physical test using the prototype was 800 millimetres by 600 millimetres. The device at its current state is able to offer the user a work area of 845 millimetres by 570 millimetres. This work area attained has an appropriate range when compared to the initial work area estimations.

From the tests performed on accuracy we were able to determine that further investigation is required. Very little joint rotation occurs since the device has large separation lengths (700 millimetres) between each potentiometer. Therefore the main area for investigation would be testing the accuracy and sensitivity of the potentiometers currently being used. The system might require a more sensitive potentiometer to be used, or require the use of a different technology such as a digital encoder.

The potentiometer currently being used has been found to have high degree of precision during the repeated tests. The main problems associated with the precision during testing were the introduction of human error and also read errors occurring from the ADC's.

During the programming of the GUI, I encountered a problem that was also associated with the accuracy of movements. When moving the physical device in a straight line it was always reflected on the computer as a curve. The use of more accurate potentiometers or a different technology could possibly eliminate this problem.

5.7 General problems

One major problem I had during the GUI development was that the cursor would always move in one direction (more vertical) followed by a second direction (more horizontal) to plot the movements of the device. I discovered that the problem was that the ADC would transfer a value and be read by the RS232COM form. This value would be processed by the position algorithm (section 4.11.2) where the position would be calculated based on this one value change.

It would then grab the second value read from the other potentiometer and plot its position, this would create a zig zag pattern as the arm was moved in a straight direction. This pattern can be seen in figure 5.5. I overcame this problem by placing the position algorithm in a loop which would only allow the location to be calculated if both the values had been updated and were relative to the current location.

Initially the amount of processing which had to occur for calculations and data processing caused the overall program to run very slow. To improve the overall performance of the program I rewrote several sections of the program which include the RS232COM form, ApplicationWindow form and the processing code on the development board. Rewriting this code allowed more computations to be performed on the development board which removed some strain from the PC. The GUI code was also significantly simplified requiring less computations to be performed.

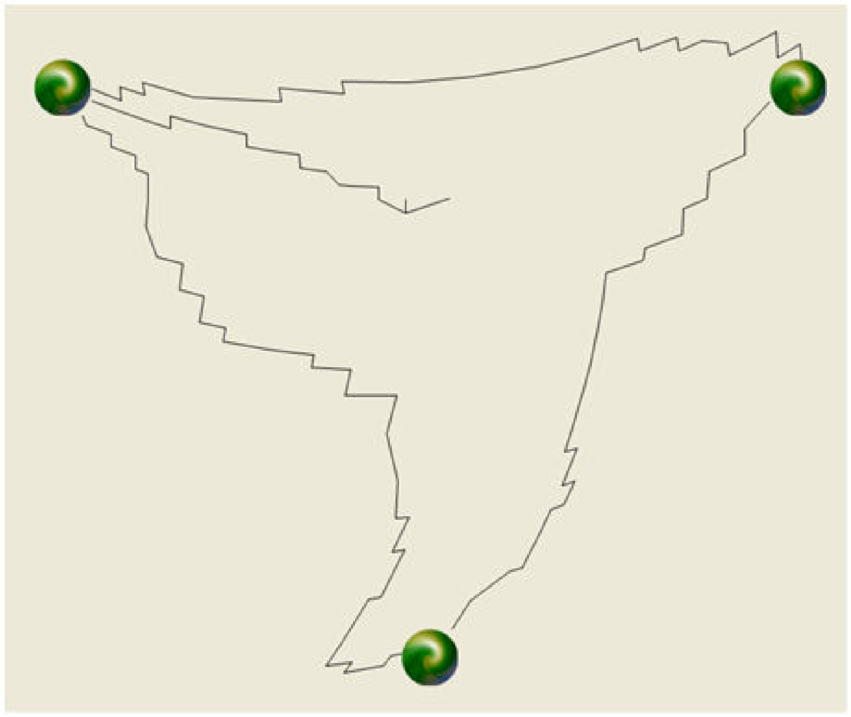


Figure 5.5: Zig Zag Pattern

There was an instability in the older version of my program which caused the system to crash. This instability was due to the “invoke” method used for reading the data from the serial port. Upon initial calibration the program would sometimes crash due to an overflow problem. To correct this problem I added the character count routine visible in figure 4.28 to clear the textbox buffer if the result exceeded eight characters. I also limited the float of the values derived from the ADC’s to two decimal places.

I have included in the design of the device a button located on top of the handle. This button is to enable/disabled the device allowing the operation of the mouse for option changes. I spent a significant amount of time investigating the use of this button on several pins available on the development board but was unable to achieve the desired result. Since I already understood the operation of the ADC’s on the development board I used a further ADC to operate the button. The use of an ADC for button detection lead to problems involving noise, which I overcame by creating a loop in the code (section 4.6). The optimal solution would be to change to a pin that is designated for impulse detection.

5.8 Incomplete Work

Due to time constraints I was unable to achieve all the requirements for the system that was specified in chapter two. The final result that I was able to attain can be seen in figure 5.6**.** This section will discuss the system requirements which were unable to be fulfilled over the duration of this project. The introduction of these additional system components would fulfill all the requirements for a fully operational physical rehabilitation technology.

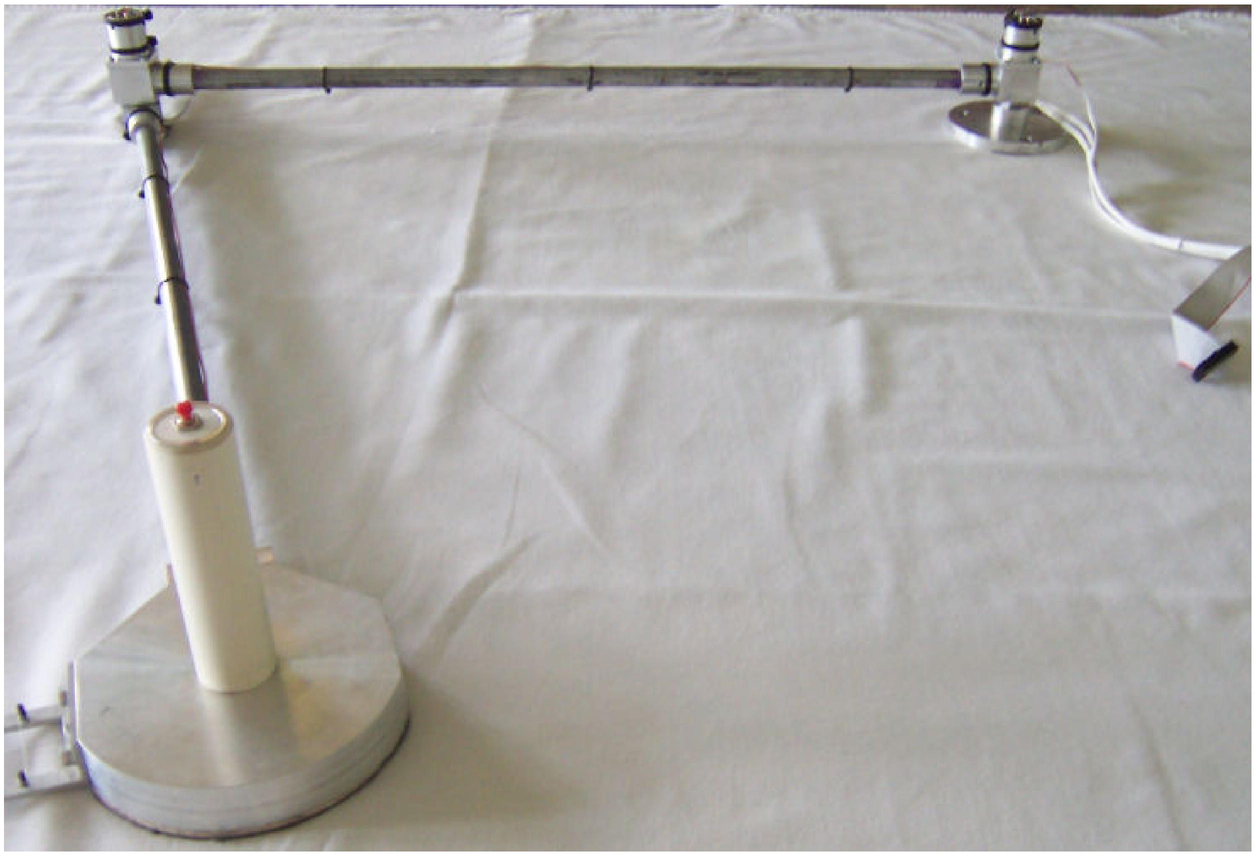


Figure 5.6: Final Device

5.2.1Hardware

The most important system component missing from the device in its current state is the technology to enable active training. Several technologies were investigated to introduce active training into the system, (section 3.4) designs were also created to implement these technologies but due to manufacturing delays I was unable to continue further development of the device and have the required components manufactured. The full listing of all technical drawings of the final manufactured design can be found in appendix B.

There are many smaller components required for a fully functional system but without the active technology incorporated some of these ideas were not able to be implemented. These include:

- A device to support the weight of the arm while the physical therapy is being performed. This would assist the patient by removing excess strain from their shoulder and bicep.
- A wrist strap that would hold the patients hand on the device. This would be used for patients having difficulty with finger strength and control.
- An emergency stop button to disable the active training at any time during the rehabilitation process.
- A special bench that would be large enough to support the full operation of the device. This bench would require a cutout section to allow the motors to be concealed underneath the bench.

5.2.2 Software

There are several advances required in the software to create a more successful GUI for users. One such advancement would be the ability for patients to have their own personal profile. The patient would be able to log into their profile and review their progress at any stage.

Currently the software has the ability to log movements made by the patient. By logging patient movements we are able to visually recreate the path the user has moved at a later date. An advancement on this current style would be to introduce smarter software to automatically calculate improvements made by the patient. This smarter software would be more user friendly allowing improvements to be displayed on a single screen as graphical plots or percentages.

To automatically rate improvements the software would take into consideration several factors which would include, movements deviating away from an idealistic line as seen in figure 5.7, if movements are changing directions (towards/away from the idealistic line) seen in figure 5.8 and weather the movements deviates across the idealistic line as seen in figure 5.9.

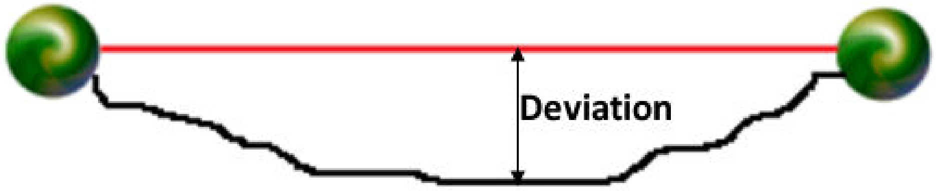


Figure 5.7: Deviation away from idealistic line

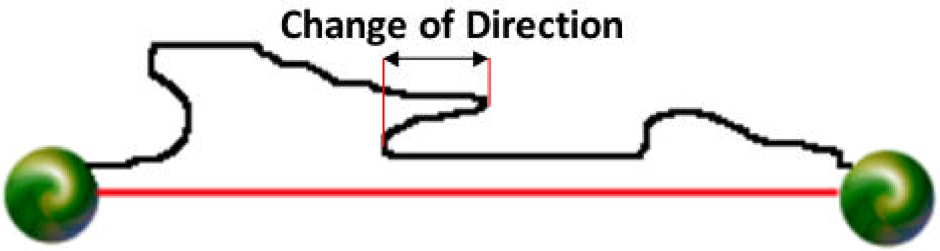


Figure 5.8: Change in direction

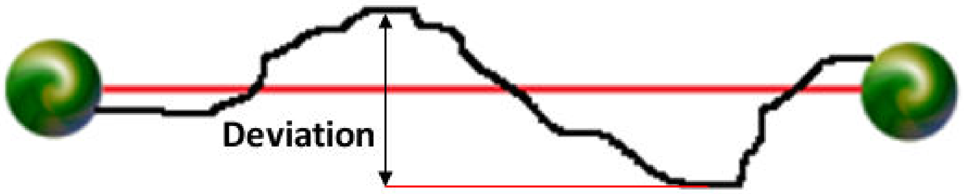


Figure 5.9: Deviation across idealistic line

5.9 Conclusion

This section has discussed the results obtained from physical tests of the device. I have derived several conclusions relating to the operational status of the final result. We have been able to see that the system was able to meet the expectations for a maximum working area specified in section 3.2.

It was found that the precision of the device was very high and in most cases the difference in results was cause by human error. In some situations there were abnormalities where the points would appear off course only to have it return quickly to the normal flow of the path. It was found that this was caused by ADC read errors and two ideas were discussed to remove the error. These ideas were, designating longer ADC query time, or to sample more values and then finding an average of the result.

The overall accuracy of the device was found to be poor. A technique of offsetting the calculated results was attempted but this technique introduced further problems into the system. It was concluded that the accuracy problems could be associated with inaccuracies of the potentiometers used.

The following section will conclude this dissertation by reviewing all the information studied during this project. It will also discuss the final results of the system created and suggest modifications to improve the system.

Chapter 6 Conclusions

6.1 Introduction

Over the duration of this project I have been faced with the task of creating a technology that would be able to be introduced into the rehabilitation system for people who suffer from hemiparesis of the upper extremities caused from a stroke.

This chapter will look at the research undertaken and the method that was used to achieve a final product. The overall results of the system will be discussed and suggestions for future work will be given.

6.2 Review: Initial Problem

This project was to design and build a technology that would be able to be introduced into a rehabilitation program with a smooth transition to assist in stroke rehabilitation. The rehabilitation program targeted stroke patients suffering from paralysis of the upper extremities. From the start of this project there were fundamental goals established which included the system must be lightweight, freely moving, affordable and be able to be easily transported to allow the technology to be operated from the comfort of a patient's home. Many patients suffer from anxiety [1] once leaving the safety of the hospital. Accomplishing the fundamental goals would allow patients to complete significant portions of their rehabilitation at home therefore this anxiety could be avoided.

Initial research was conducted to understand the technical details of strokes. From this research I was able to understand

- What occurs during a stroke
- The different kinds of strokes
- The important measures required to be carried out after the occurrence of a stroke.

It has been found that strokes are associated with several lifestyle circumstances. These could include old age, an unhealthy diet, smoking or head injuries.

Once able to understand the background information about strokes I conducted a literary review on the current methods and experimental technologies used in the rehabilitation process. It was found that the techniques currently being used involved completing tasks that would be required in everyday living (e.g. dressing oneself, threading a needle, making a coffee). Some methods were more physical than others, such as the constraint induced methods. Several technologies were found to be introduced into experimental rehabilitation programs. These technologies included AutoCITE and T-WREX.

Using the knowledge obtained from the literary review I was able to design several systems that would be able to handle all the requirements for stroke rehabilitation. From this, one system was chosen and a prototype was constructed. Once the prototype was constructed and design finalized the device was built.

After the device was built I was able to create a GUI to handle the job of tracking and logging movements. The initial design involved construction of a bench for the device to operate which had LED lights under the surface. These LED lights would be used to establish a goal point to locate the handle of the device. This was replaced with a simple game designed in the GUI.

6.3 Dissertation Review

There are many techniques currently being used to help rehabilitate stroke patients, some of these techniques involve simple exercises such as inserting blocks into a pegboard or to thread a piece of string through a hole (section 2.3). Some techniques currently being used are more intensive such as constraint induced rehabilitation (section 2.4). Constraint induced rehabilitation involves the patient having their able limb restrained for up to 90% of their daily activities, thus forcing them to develop control over their paralyzed limb. This method has been seen to be very successful for allowing patients to develop motor control. From the research that has been conducted the introduction of robotic-aided neuro-rehabilitation has outperformed all other techniques [21]. Results from studies performed have shown that patients are able regain significant motor control with as little as 25% contact time with a physiotherapist [15].

Researching various styles of rehabilitation has shown that although techniques might vary they all aim to teach the patient three things associated with motor control. These are to increase the patient’s perception, coordination and muscle strength.

By introducing technology into the rehabilitation system we would be able to offer patients benefits over the current methods being used. Such benefits as, a computer based technology that is able to repeat tasks endlessly with the precision of movements the same each use. The technology would not suffer from fatigue which is often associated with human dependant rehabilitation.

Since the technology is computer based the movements made by the patient are able to be recorded accurately, this allows the information to be recalled for progress reports as required and allowing exercise routines to be constantly modified to suit the patient's needs. Using the computer technology further we are able also remove the mundane exercises associated with physical rehabilitation by masking it behind a video game.

After developing several concepts on paper the designs were introduced into Pro/Engineer Wildfire (a solid modeling application) to allow the ideas to be seen more clearly (section 3.3). This allowed faults with the system to be located before further investigation into the design would continue. A design was chosen dependant on several factors which included, the safety of the device, simplicity, able to provide all the requirements for rehabilitation, ability to easily relocate the device and affordability.

The ability to introduced active movements into the device was the final requirement of the system (section 3.4). Many technologies were investigated to allow this function to be added. These technologies include:

- Linear Pneumatics
- Rotary Pneumatics
- Air Muscles
- DC Motors

The technology which was most logical to implement into the system was the DC motor design seen previously in figure 3.16. This system offered more benefits than any other system. Before designs were finalized for construction a prototype was created allowing a physical example to be tested to see if it met all system requirements (section 4.2). The designs were then finalised and device constructed.

GUI software was developed to be used in coordination with the technology (section 4.10). This software had many options introduced into it to make it as versatile as possible for all users. The ability to log and reproduce movements was also added.

Once the device had been constructed and GUI developed they were connected together using a development board (LPC-MT-2138). The development board allowed the analogue signals derived from the potentiometers (used for angle calculations) to be converted into a digital signal which the GUI software was able to interpret.

Once the technology was successfully running I was able to test the device and analysis the results from the system (section 5.6). I found that the system had very high precision and was able to attain the range-of-motion which was required for sufficient exercise. However the accuracy of the system did not perform as well as expected. Further investigation into what is causing these inaccuracies is required. One possibility causing the poor accuracy could be purely related to the quality of the potentiometers used in the device.

6.4 Future Work

There is still some remaining work which needs to be done before this device can proceed to the following stage of performing physical testing on stroke patients. The remaining work required consists of further work on both the hardware and software. This future work is discussed in the follow sections.

6.4.1 Hardware

The introduction of technology to allow active assistance/resistance during operation is the most important necessity currently required. There were several systems discussed in the conceptual designs section that would allow this to be introduced. Of these designs it was found that the most logical system to fulfill the fundamental requirements was the DC motor design.

A system that does not support active movements but still assists the patient could be introduced. An example of this system can be seen in figure 6.1. This design consists of a stepper motor, a freely spinning wheel and the structure housing. This system would be bolted in the location between the handle of the device and the extension bar.

This system is designed so that the stepper motor would adjust and lock the freely rotating wheel to point in the direction of the target. The patient would still have to move the entire device under their own strength. The directional guidance will assist in teaching muscle coordination and control.

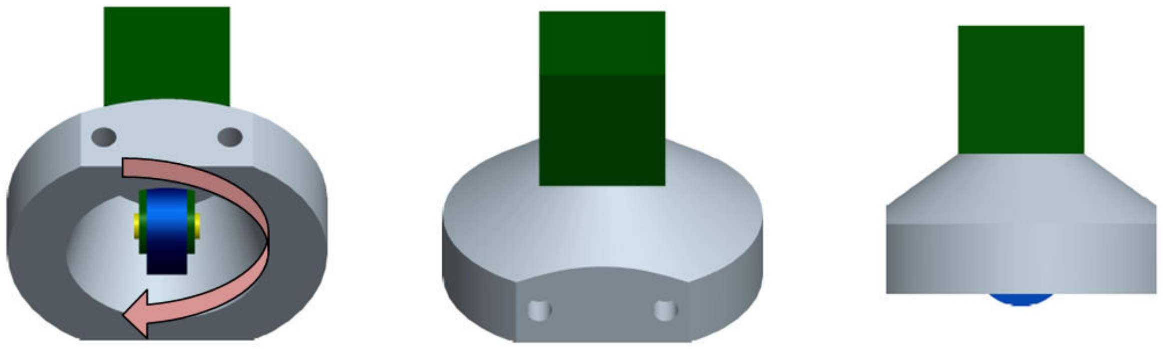


Figure 6.1: Directional Wheel

The problems currently associated with accuracy (discussed in section 5.5) in the system are thought to be caused by the quality of the potentiometers. Further investigation is required before another technology of higher quality is invested in. One such technology that could be introduced into the system to correct this problem is a digital encoder.

The requirements of the system have directly resulted in the overall system requiring a large work area. Therefore it has become a necessity for the system to have a special workbench created to house the device. The requirements of the workbench are:

- Low friction surface. e.g. glass
- Enable full extension of the device in all appropriate directions.
- Provide room to house the motors (if DC motor design is chosen).
- Include LED lights under the surface layer to assist in directional guidance (optional).
- Enable the end to be attached quickly and safely.

To assist in the support of the patient limb two apparatus's are required for the system. The first apparatus is a hand strap to assist the patient holding the free end of the device. The other apparatus is a system to support the weight of the arm. This apparatus could be attached to the bench or chair used during the operation of the device. Ultimately it should support the arm before the elbow joint as to eliminate restrictive movements.

An emergency button should be installed once the device has reached a stage of being active. This button would disable the technology instantly thus helping to eliminate injuries from occurring. Lastly torque sensors could be introduced into the system to determine muscle strength. Torque sensors would only be able to be used during zero-force movements otherwise strength values would be incorrect.

6.4.2 Software

There are several advancements required for the software to be flexible for daily use. One requirement is the introduction of individual profiles for each of the patient's who use the device. Profiles would allow patients to have private logging of performed exercise sessions. This would eliminate confusion between different patient's session logs, and also allows the information to remain confidential between themselves and the physiotherapist. Individual profiles would also allow for personal system requirements to be defined such as the maximum allowable torque provided. By doing this an additional measure has been introduced into the system for the patient's safety.

A major software improvement would be the creation of smart algorithms to automatically rate the patient's accuracy, precision, muscle strength and overall progress. Progress would be able to be displayed as either percentage ratings or visually by using histogram plots. The use of this smart software would allow faster patient progress reports, allowing exercise routines to be changed more often to suit the patient's needs. It would also allow the patient to review their own progress without requiring detailed explanations of how the movements are gauged for improvements.

6.5 Conclusion

From the research undertaken during the process of this project there have been many important issues raised when comparing human dependant rehabilitation against robot-aided rehabilitation. A computer based technology has the ability to reproduce the required assistive/resistive movements with the same predetermined idealistic characteristics for each time of use. Whereas if these same active movements were provided by a human, inaccuracies in movements will eventually occur as muscle fatigue and boredom occurs.

Introducing technology also allows the movements made by patients to be logged and reproduced perfectly to allow the most beneficial exercise program to be developed. Implementing robotic technology into the rehabilitation process would allow the patient to exercise without requiring long durations of expensive personal physiotherapy. This allows the patient to save money on the rehabilitation process and also allows the physiotherapist to work with more patients.

Research has shown that patients who have undergone robot-aided neuro-rehabilitation or have used this technology for parts of their rehabilitation have shown results that outperform human dependant rehabilitation which currently dominates the physical rehabilitation process.

The videogame industry is rapidly expanding, last year the United States alone made over $12.5 billion, and shows no signs of slowing down [31]. This demonstrates the popularity that video games have in current society. By using a computer based technology we are able to introduce video games into the rehabilitation process allowing the mundane exercises to be masked with entertainment.

As the human resources in the medical field dwindle due to being underfunded [32] the step towards introducing medical robotics for rehabilitation is becoming more necessary. This device is a step forward for medical rehabilitation and will allow patients to regain more control of their effected limbs than other human dependant techniques. This device is designed for the good of mankind to help those people less fortunate to be able to receive the same quality of rehabilitation as anyone else.

References

[1] J Hall
Stroke of bad luck (2007) (online)
http://www.news.com.au/entertainment/story/0,26278,22103787-5007197,00.html
Last accessed 07/October/2008

[2] Heinemann Australian dictionary
4th edition (1992)

[3] BBC News: Heart disease & stroke: the facts (online)
http://news.bbc.co.uk/hi/english/static/in_depth/health/2000/heart_disease/stroke.stm
Last accessed 07/October/2008

[4] D. Jensen
Strokes can happen at any age (online)
http://shamvswham.blogspot.com/2008/07/strokes-can-happen-at-any-age.html
Last accessed 07/October/2008

[5] S. Cohen
Preventing recurrent ischemic stroke: A 3-step plan; to prevent recurrent stroke, address the patient's risk factors, clear stenosis, and thin the blood (online)
http://findarticles.com/p/articles/mi_m0689/is_/ai_n13795388
Last accessed 07/October/2008

[6] V Howard, PhD; G Howard, PhD
'Nonmodifiable' risk factors for stroke: age, race, sex, and geography (online)
http://books.google.com.au/books?id=NCzLa5B7PRcC&pg=PA11&lpg=PA11&dq=stroke+age+40%25&source=web&ots=laaWj0yBFw&sig=qZqv934srN3nIT6j47lajSF1Vgo&hl=en&sa=X&oi=book_result&resnum=2&ct=result#PPA12,M1
Last accessed 07/October/2008

[7] P Langhorne, PhD, FRCP; D Stott, MD, FRCP; L Robertson, RGN; J MacDonald, FRCP; L Jones, RGN; C McAlpine, FRCP; F Dick, RGN; G Taylor, BSc; G Murray, PhD
Medical Complications After Stroke (online)
http://stroke.ahajournals.org/cgi/content/abstract/31/6/1223
Last accessed 07/October/2008

[8] Post-Stroke Rehabilitation Fact Sheet (online)
http://www.ninds.nih.gov/disorders/stroke/poststrokerehab.htm
Last accessed 07/October/2008

[9] S Housman; V Le; T Rahman; R Sanchez; D Reinkensmeyer
Arm-Training with T-WREX After Chronic Stroke: Preliminary Results of a Randomized Controlled Trial (online)
http://www.smpp.northwestern.edu/Robotics/pubs/TWREX_ICORR_2007_revised_final.pdf
Last accessed 07/October/2008

[10] Cheap Technology for Better Stroke Rehab (online)
http://medgadget.com/archives/2007/12/cheap_technology_for_better_stroke_rehab.html
Last accessed 07/October/2008

[11] Definition of Perception (online)
http://www.sapdesignguild.org/resources/optical_illusions/intro_definition.html
Last accessed 09/October/2008

[12] Stroke Rehabilitations (online)
http://www.strokefoundation.com.au/component/option,com_docman/Itemid,128/task,doc_view/gid,104/
Last accessed 09/October/2008

[13] J. Desrosiers; D. Bourbonnais; H. Corriveau; S. Gosselin; G. Bravo
Effectiveness of unilateral and symmetrical bilateral task training for arm during the subacute phase after stroke: a randomized controlled trial
Clinical Rehabilitation
2005. 581-593

[14] N Bonifer; K Anderson
Application of Constraint-Induced Movement Therapy for an Individual with Severe Chronic Upper-Extremity Hemiplegia
Physical Therapy. Volume 83. Number 4
2003. 384-398

[15] H Nakayama; HS Jorgensen; HO Raaschou; TS Olsen
Recovery of upper extremity function in stroke patients: The Copenhagen Stroke Study.
Arch Phys Med Rehabil.
1994. 75:394-8.

[16] B. Lindmark
The improvement of different motor functions after stroke.
Clinical Rehabilitation.
1988. 2:275-83.

[17] A Siebers; U O'Berg; E Skargren
Advances in Physiotherapy.
2006. 8:146-153

[18] E. Taub, PhD; P. Lum, PhD; P. Hardin, MS; V. Mark, MD G. Uswatte, PhD
Automated Delivery of CI Therapy with Reduced Effort by Therapists (online)
http://stroke.ahajournals.org/cgi/content/full/36/6/1301
Last accessed 10/October/2008

[19] Innovative Stroke Rehab Therapy Takes Next Step — Automation (online)
http://main.uab.edu/show.asp?durki=80167
Last accessed 10/October/2008

[20] J. Underwood; P.C Clark; S. Blanton; D.M. Aycock; S.L. Wolf
Pain, fatigue, and intensity of practice in people with stroke who are receiving constraint-induced movement therapy.
Physical Therapy. Volume 86. Number 9
2006. 86:1241-1250

[21] H.I. Krebs, PhD; B.T. Volpe, MD; M.L. Aisen, MD; N. Hogan, PhD.
Journal of Rehabilitation Research and Development Vol. 37 No.6,
November/December 2000. Pages 639-652

[22] M. Finley, PT, PhD; S. Fasoli, ScD; L. Dipietro, PhD; J. Ohlhoff, BA; L. MacClellan, MSPH; C. Meister, OTR; J. Whitall, PhD; R. Macko, MD; C. Bever Jr, MD; H. Krebs, PhD; N. Hogan, PhD.
Short-duration robotic therapy in stroke patients with severe upper-limb motor impairment.
Journal of Rehabilitation Research and Development Volume 42. Number 5
2005. 683-692

[23] J. Patton; M. Stoykov; M. Kovic; A. Ferdinando; M. Ivaldi
Evaluation of robotic training forces that either enhance or reduce error in chronic hemiparetic stroke survivors
2006. Research Article

[24] Bi Technologies (online)
http://au.farnell.com/1520667/passives/product.us0?sku=bi-technologies-6187r5kl1-0&_requestid=140198
Last accessed 11/October/2008

[25] LPC-MT-2138 Development Board (online)
http://www.olimex.com/dev/lpc-mt-2138.html
Last accessed 11/October/2008

[26] The .NET Framework (online)
http://windows-programming.suite101.com/article.cfm/the_net_framework
Last accessed 11/October/2008

[27] Word Net (online)
http://wordnet.princeton.edu/perl/webwn?s=accuracy
Last accessed 12/October/2008

[28] Image: High accuracy Low precision (online)
http://en.wikipedia.org/wiki/Image:High_accuracy_Low_precision.svg
Last accessed 11/October/2008

[29] Word Net (online)
http://wordnet.princeton.edu/perl/webwn?s=precision&o2=&o0=1&o7=&o5=&o1=1&o6=&o4=&o3=&h=00
Last accessed 11/October/2008

[30] Image: High precision Low accuracy (online)
http://en.wikipedia.org/wiki/Image:High_precision_Low_accuracy.svg
Last accessed 11/October/2008

[31] The Video Game Explosion: A History from PONG to PlayStation and Beyond (online)
http://www.amazon.com/Video-Game-Explosion-History-PlayStation/dp/031333868X
Last accessed 15/October/2008

[32] The Health Workforce (online)
http://www.pc.gov.au/__data/assets/pdf_file/0018/10377/sub065.pdf
Last accessed 11/October/2008

University of Southern Queensland
Faculty of Engineering and Surverying

The Development of a Computer Based System for the Rehabilitation of Arm Movements of Stroke Patients

A dissertation submitted by

Nathan Prasser

in fulfillment of the requirements of

Courses ENG4111 and ENG4112 Research Project

towards the degree of

Bachelor of Engineering (Mechatronic)

Submitted: October, 2008

Volume 2

Appendix A: Project Specification

University of Southern Queensland

FACULTY OF ENGINEERING AND SURVEYING

ENG 4111/4112 Research Project

PROJECT SPECIFICATION

FOR:	Nathan Prasser
TOPIC:	Design and Build a computer based system to assess and exercise the arm movements of stroke patients
SUPERVISORS:	Dr Selvan Pather
ENROLMENT:	ENG 4111 – S1, 2008 ENG 4112 – S2, 2008
PROJECT AIM:	This project aims to design, build and program a system that will allow stroke patients to rehabilitate themselves. Requirements of the system are that the system remains lightweight for easy of transport, affordable, simple to setup and operate.

PROGRAMME: Issue A, January 2008

1) Research the cause and underlying physical effects experienced after a stroke has occurred.
2) Research current physiotherapy techniques that are being used to combat loss of limb control.
3) Research current technologies which are currently being used to assist in rehabilitation.
4) Design and build a model which will be able to move in any point within two planes of movement.
5) Do initial programming of arm to calculate arm location.
6) Modify arm to be able to provide resistance and assistance.
7) Program arm to provide resistance and assistance.
8) Design and build a bench for the arm to operate on with LED's embedded to create objectives for user to follow.

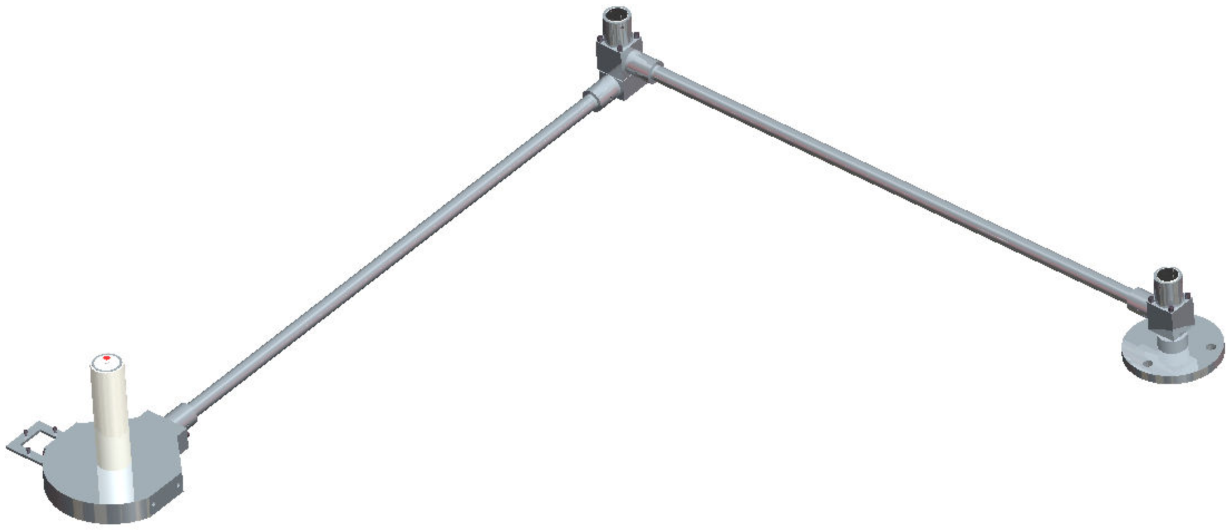
AGREED:

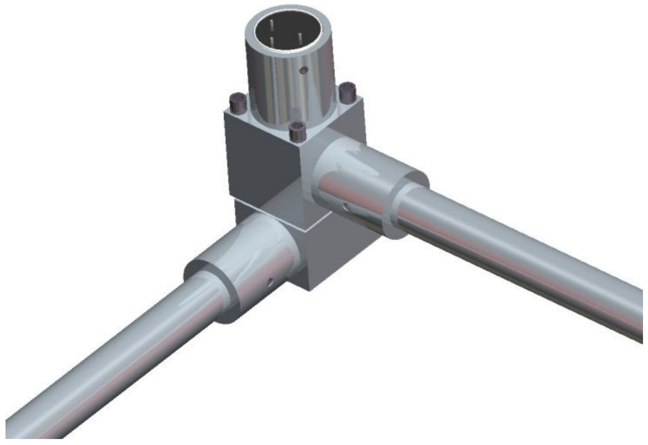
_______________________ (Student) _______________________ (Supervisor)

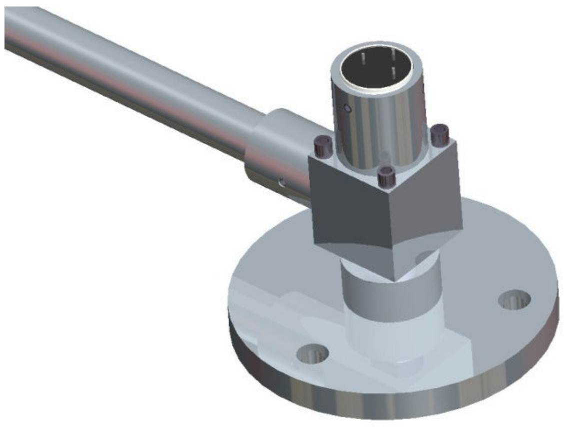
____ / ____ / 2008 ____ / ____ / 2008

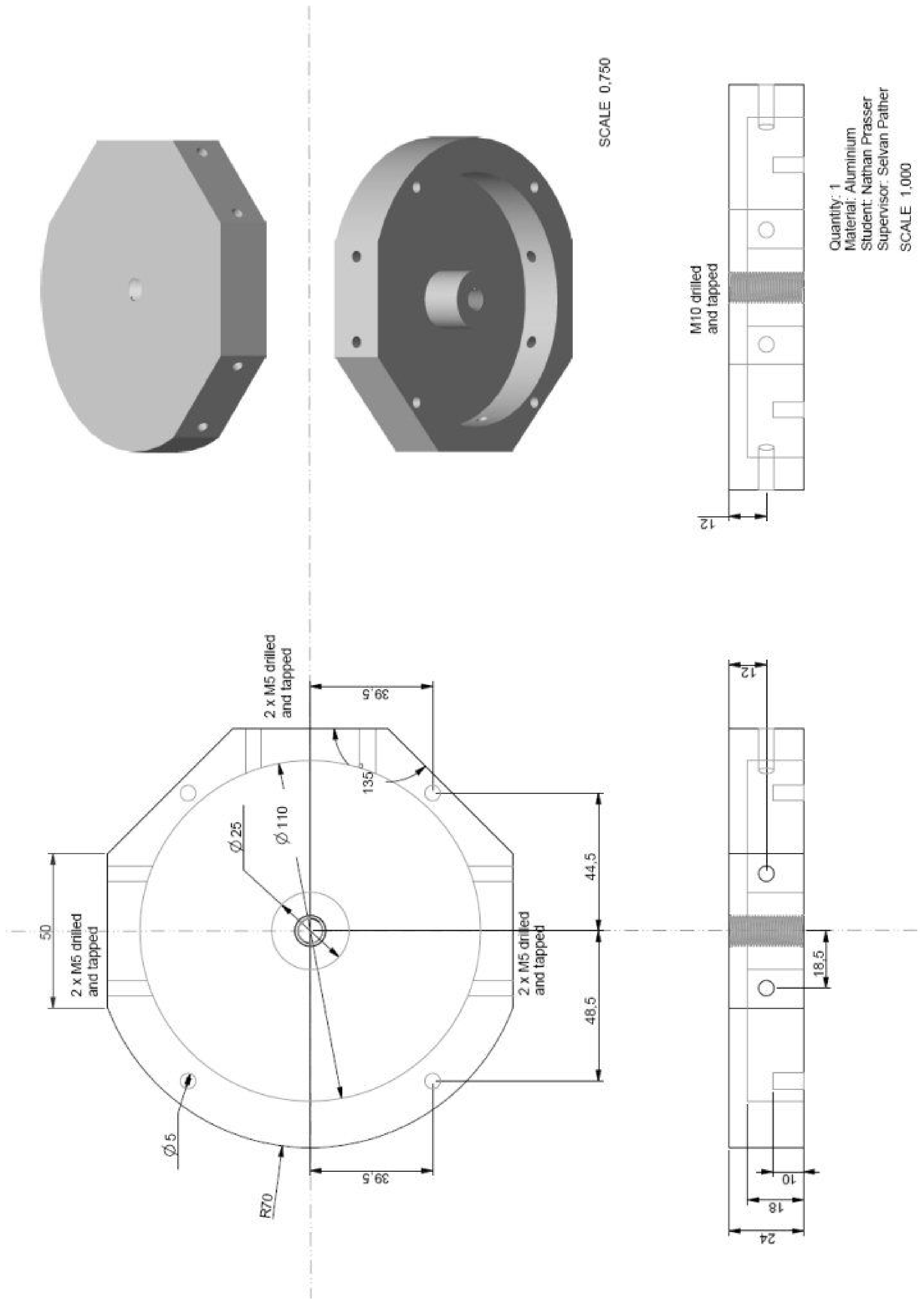
Examiner/Co-examiner: ____

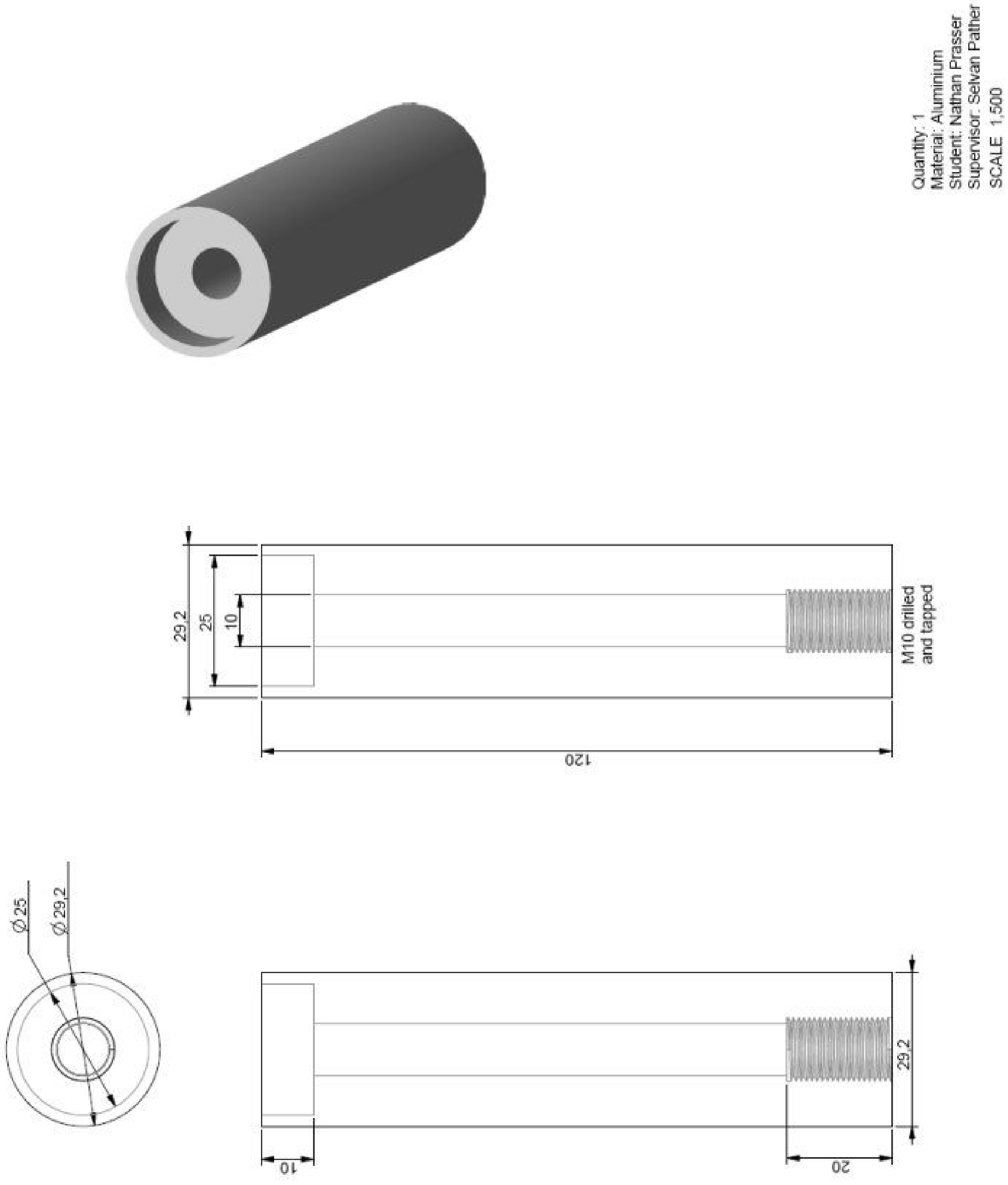
Appendix B: Technical Drawings

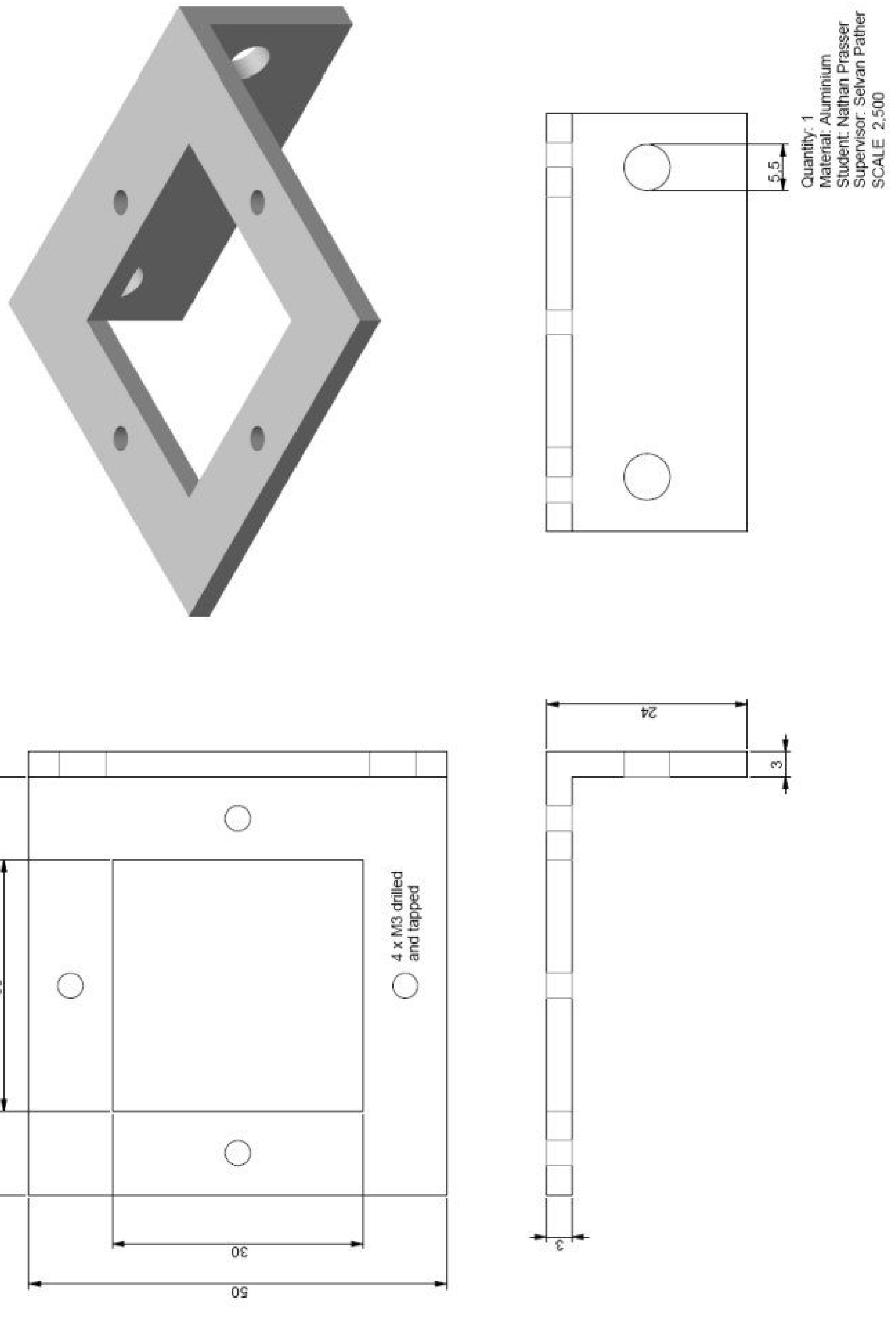


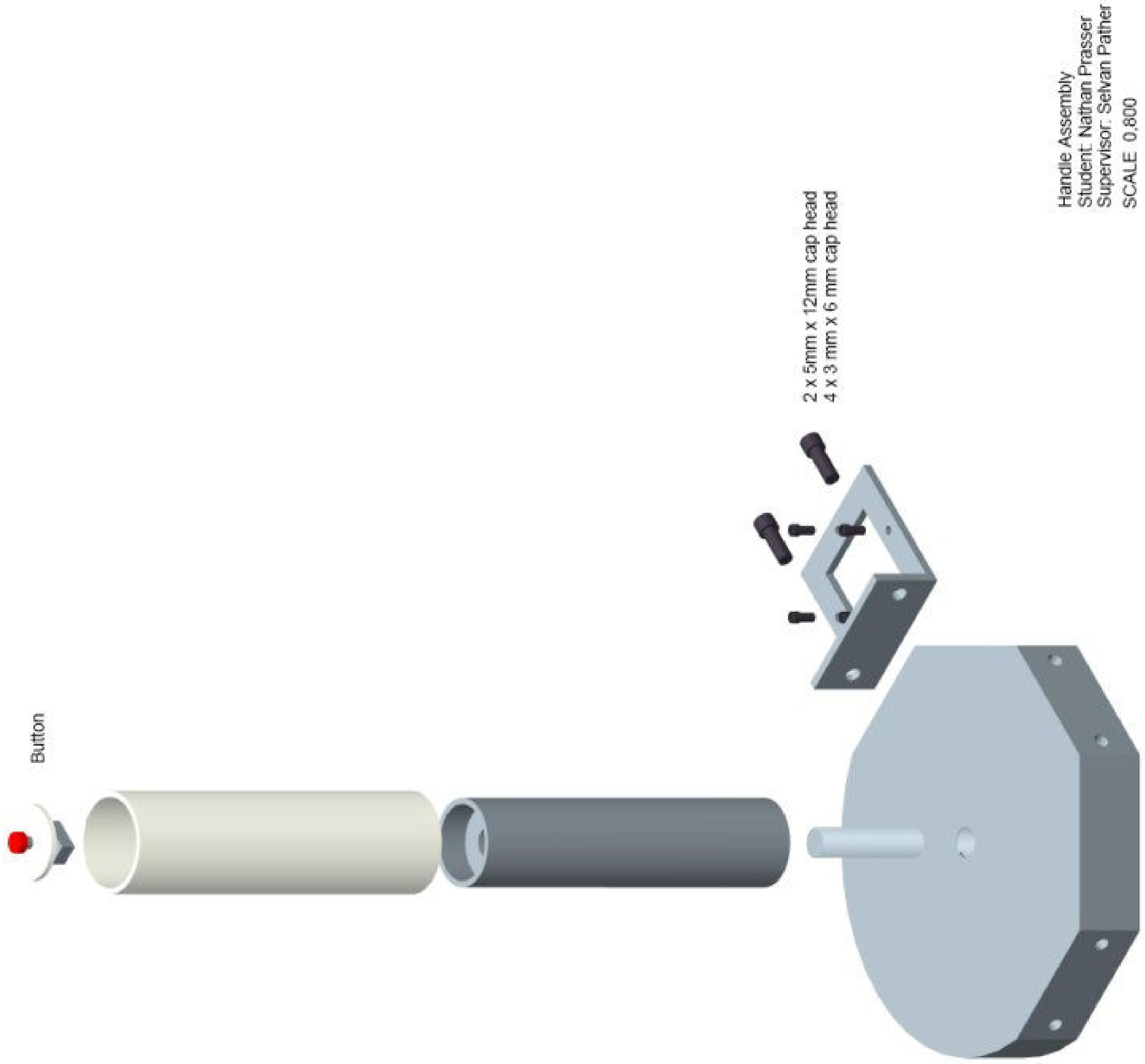


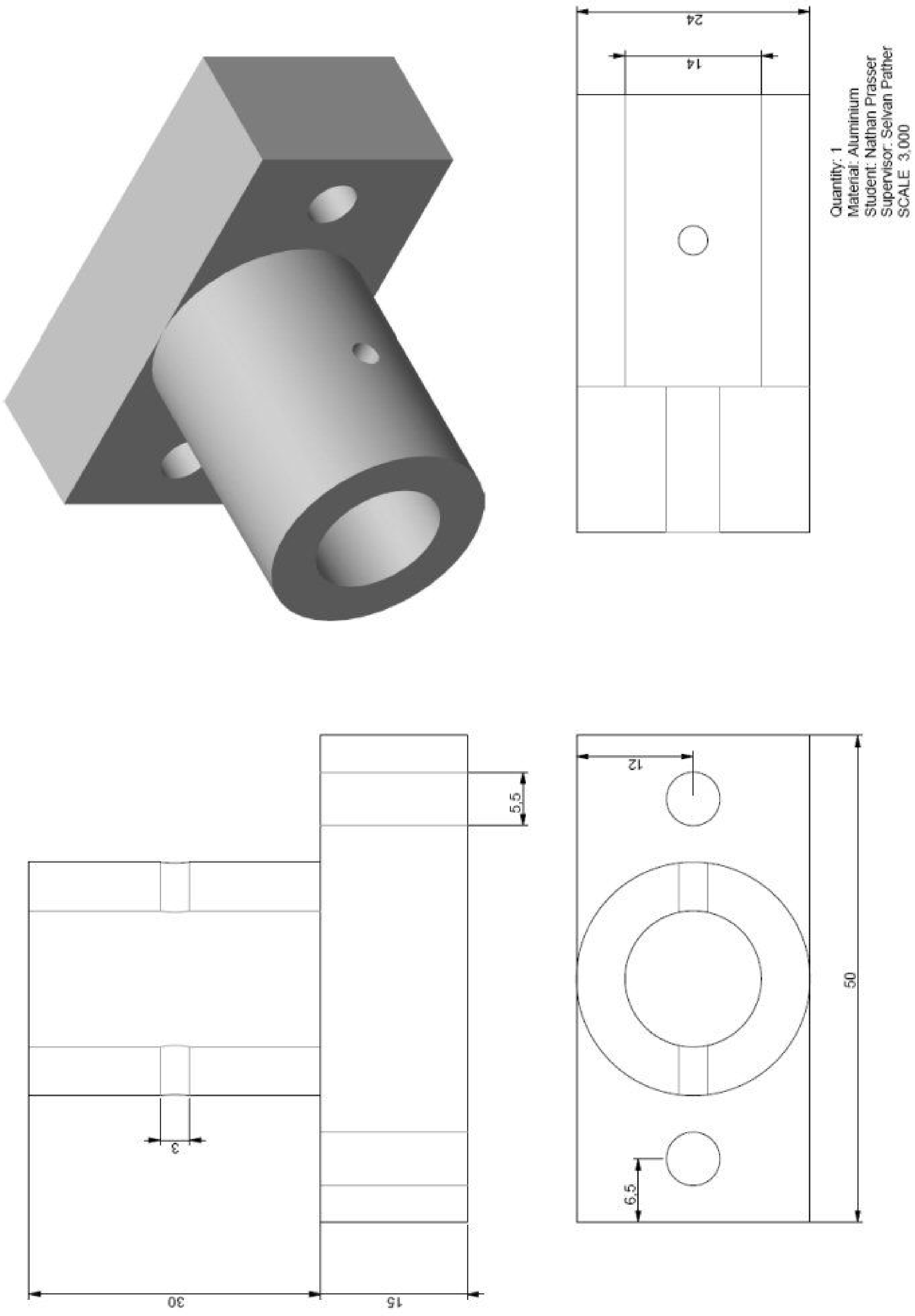


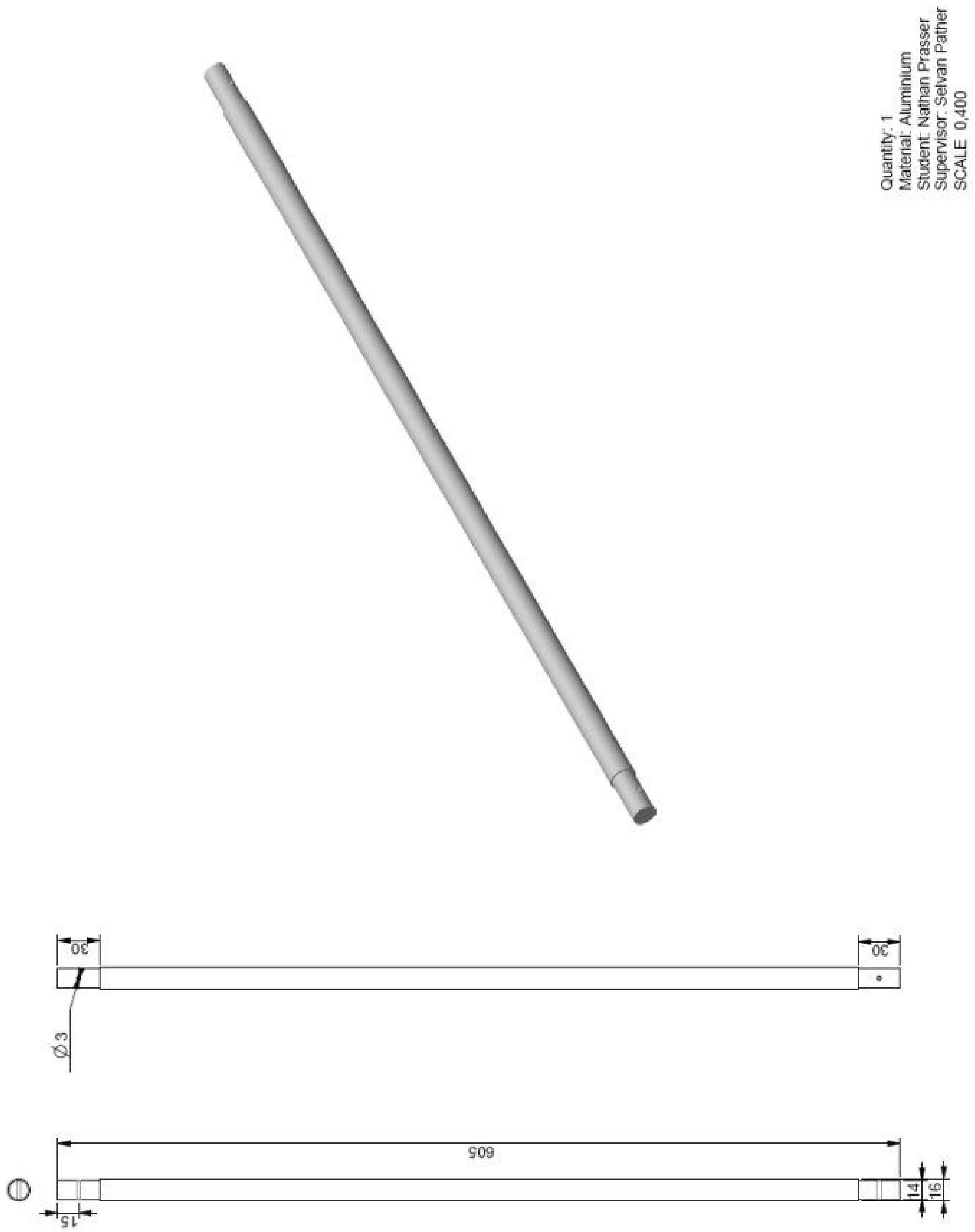

SCALE 0,750
M10 drilled and tapped
Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 1,000
12
2 x M5 drilled and tapped
39,5
135°
Ø 110
Ø 25
50
44,5
48,5
18,5
Ø 5
R70
10
18
24


Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 1,500
29,2
25
10
M10 drilled and tapped
120
Ø25
Ø29,2
29,2
10
20

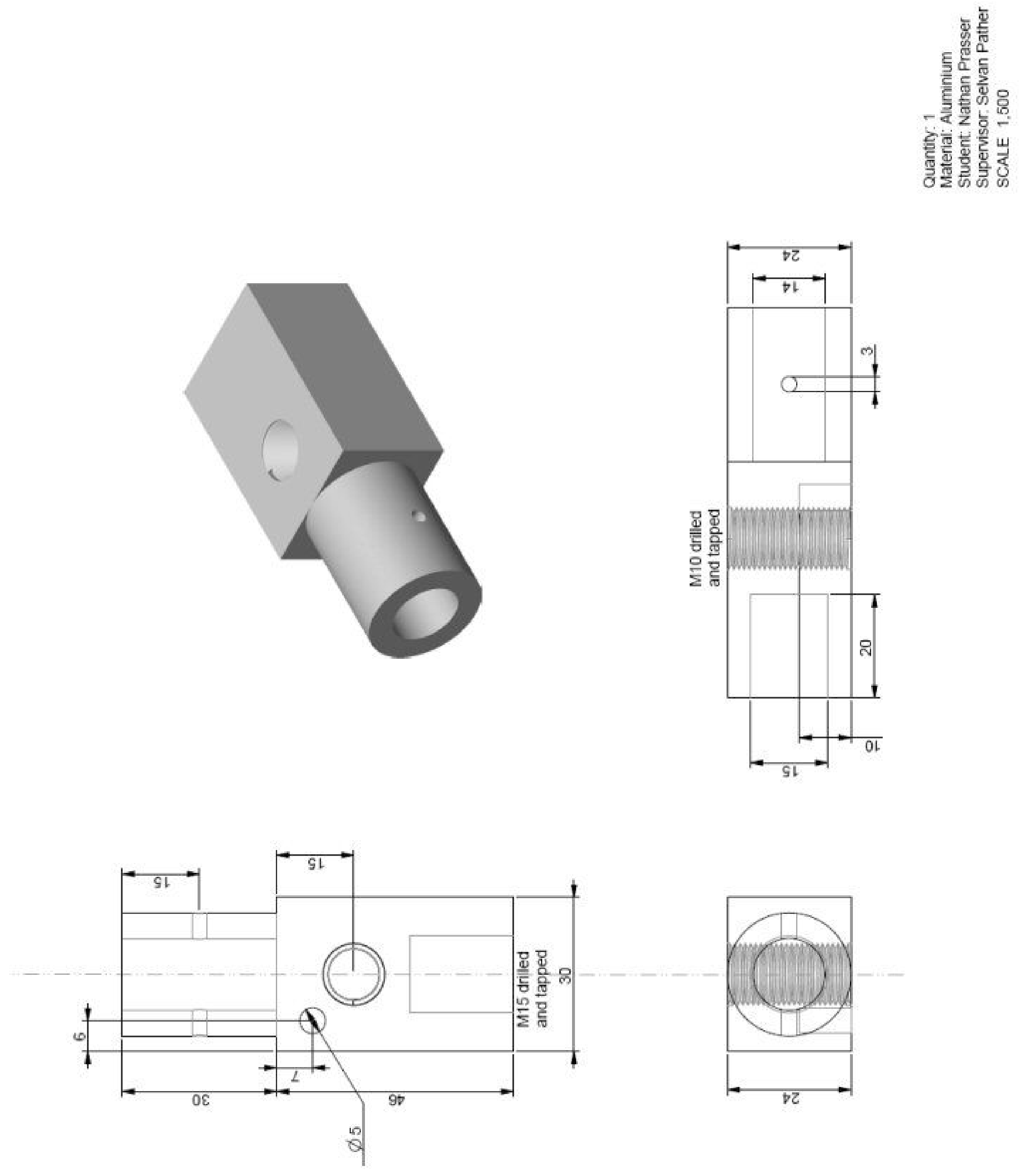

Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 2,500
5,5
24
3
50
30
4 x M3 drilled and tapped
30
50
3

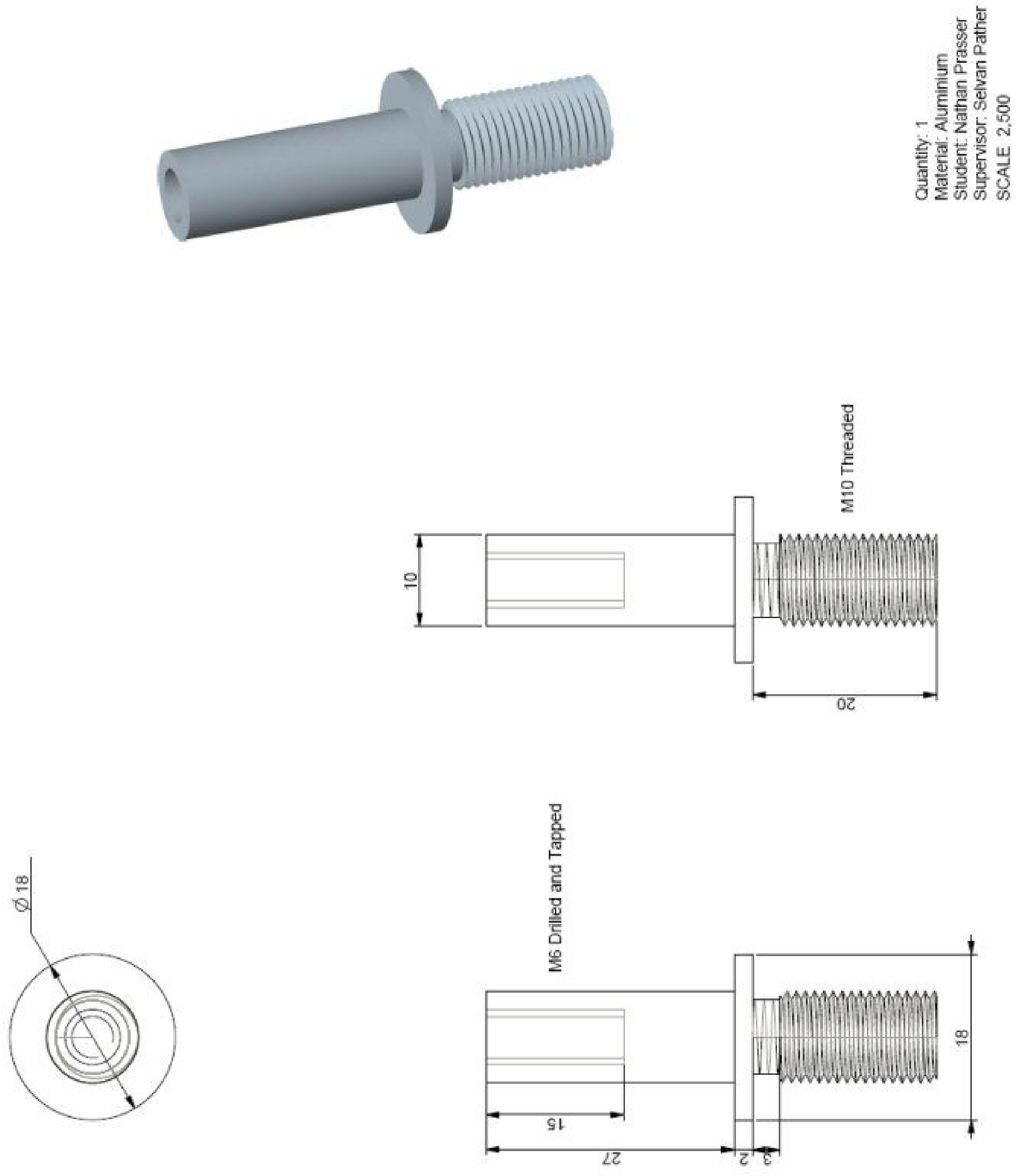

Handle Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 0,800
2 x 5mm x 12mm cap head
4 x 3 mm x 6 mm cap head
Button

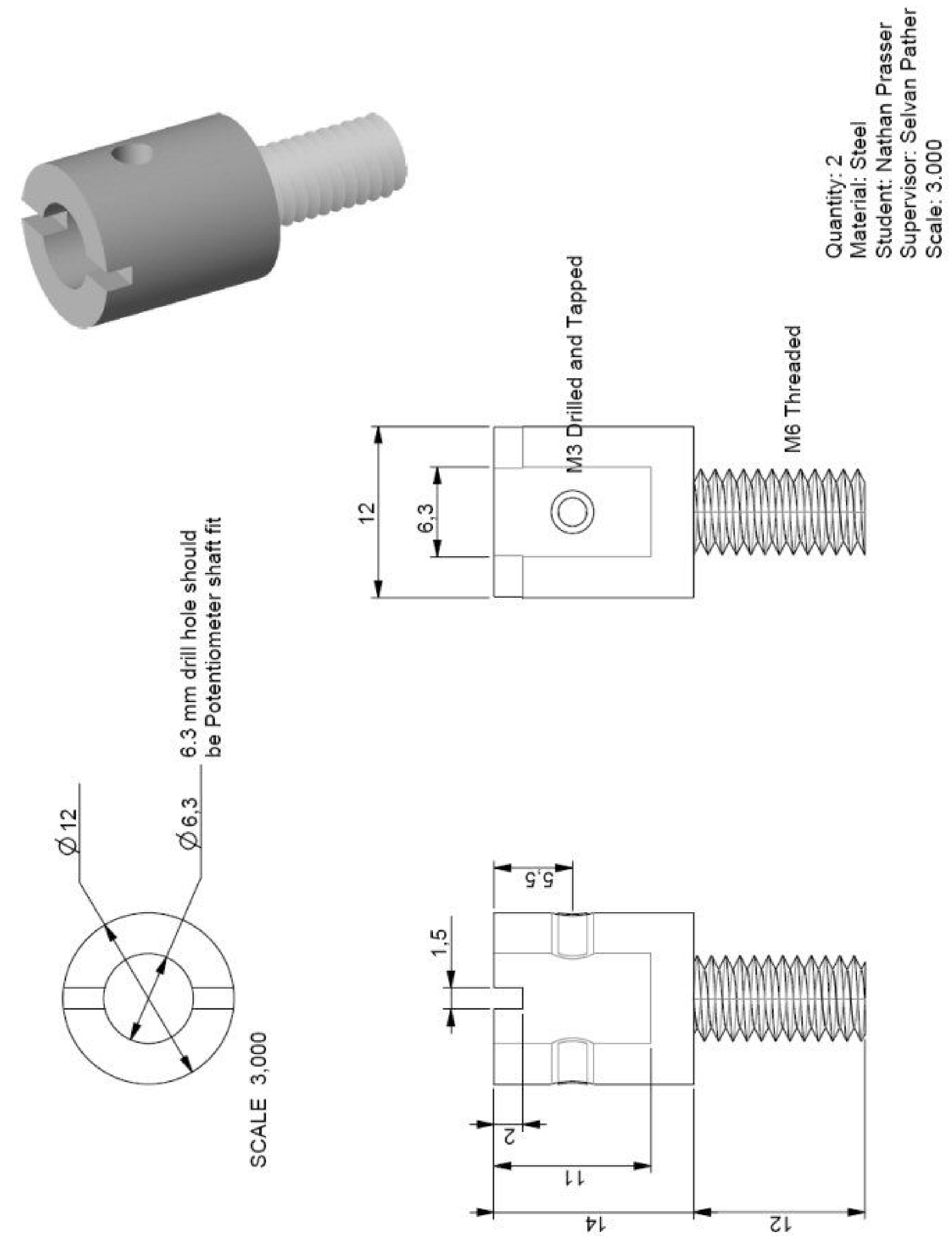

24
14
Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 3,000
12
5,5
50
3
6,5
30
15

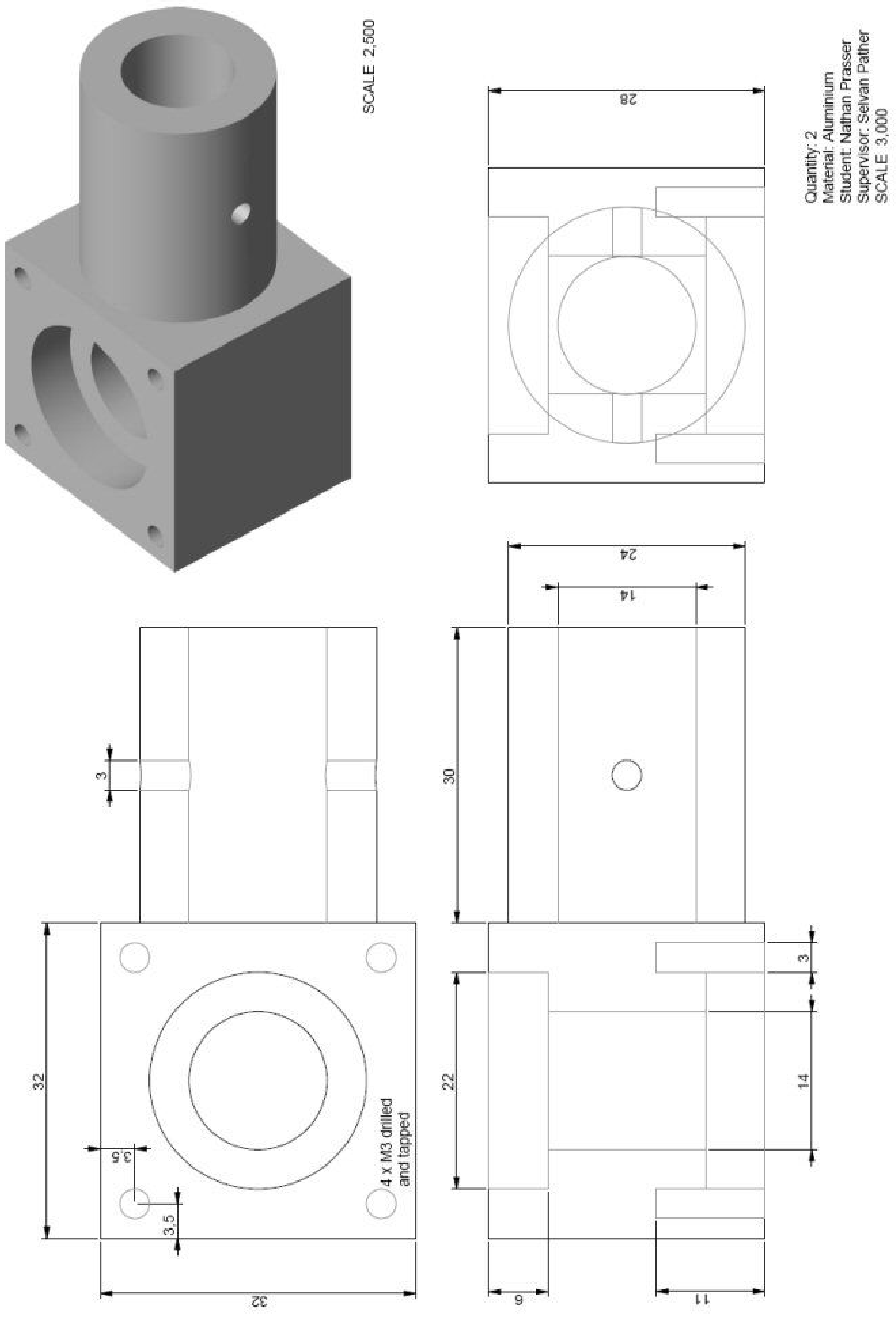


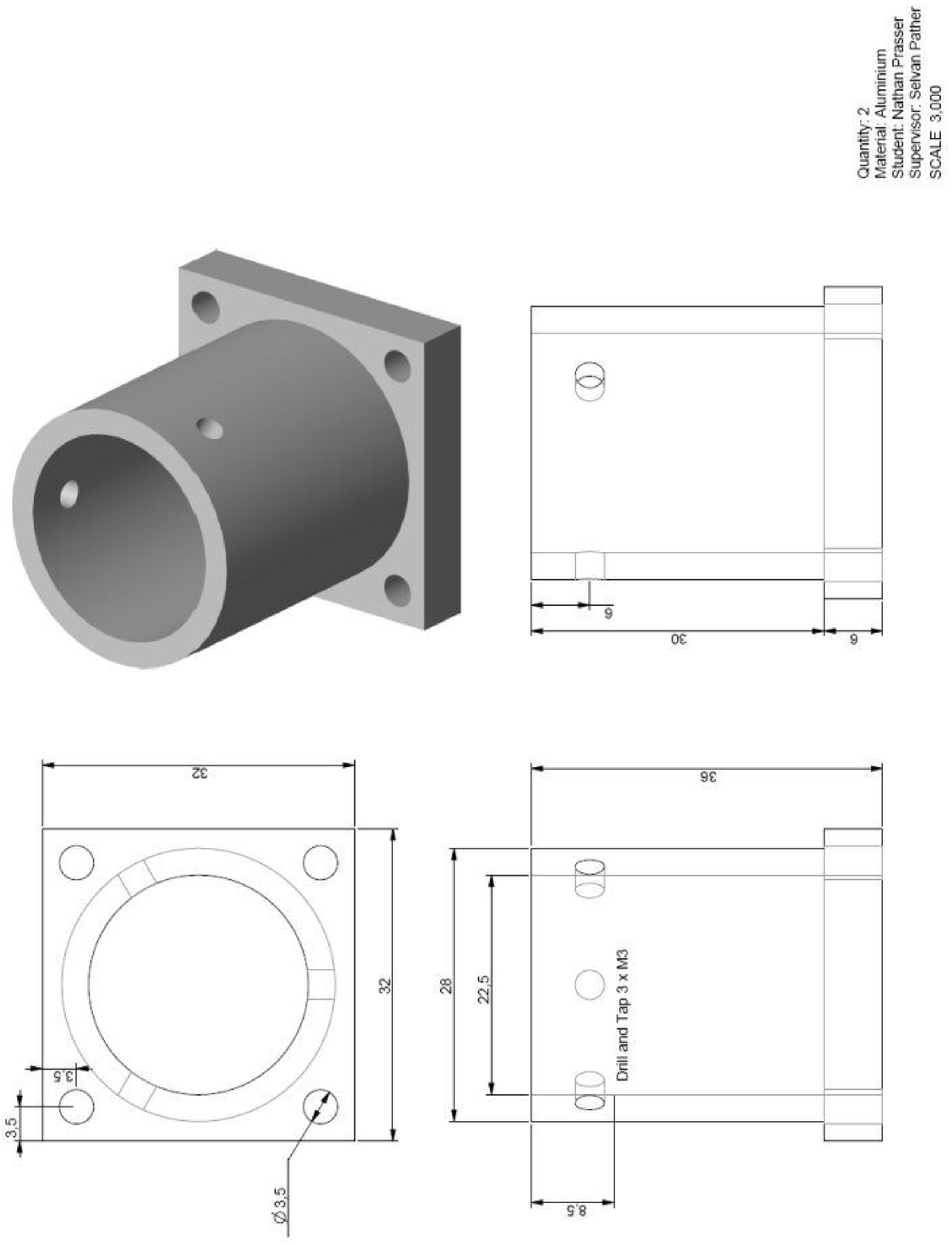
Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 0,400

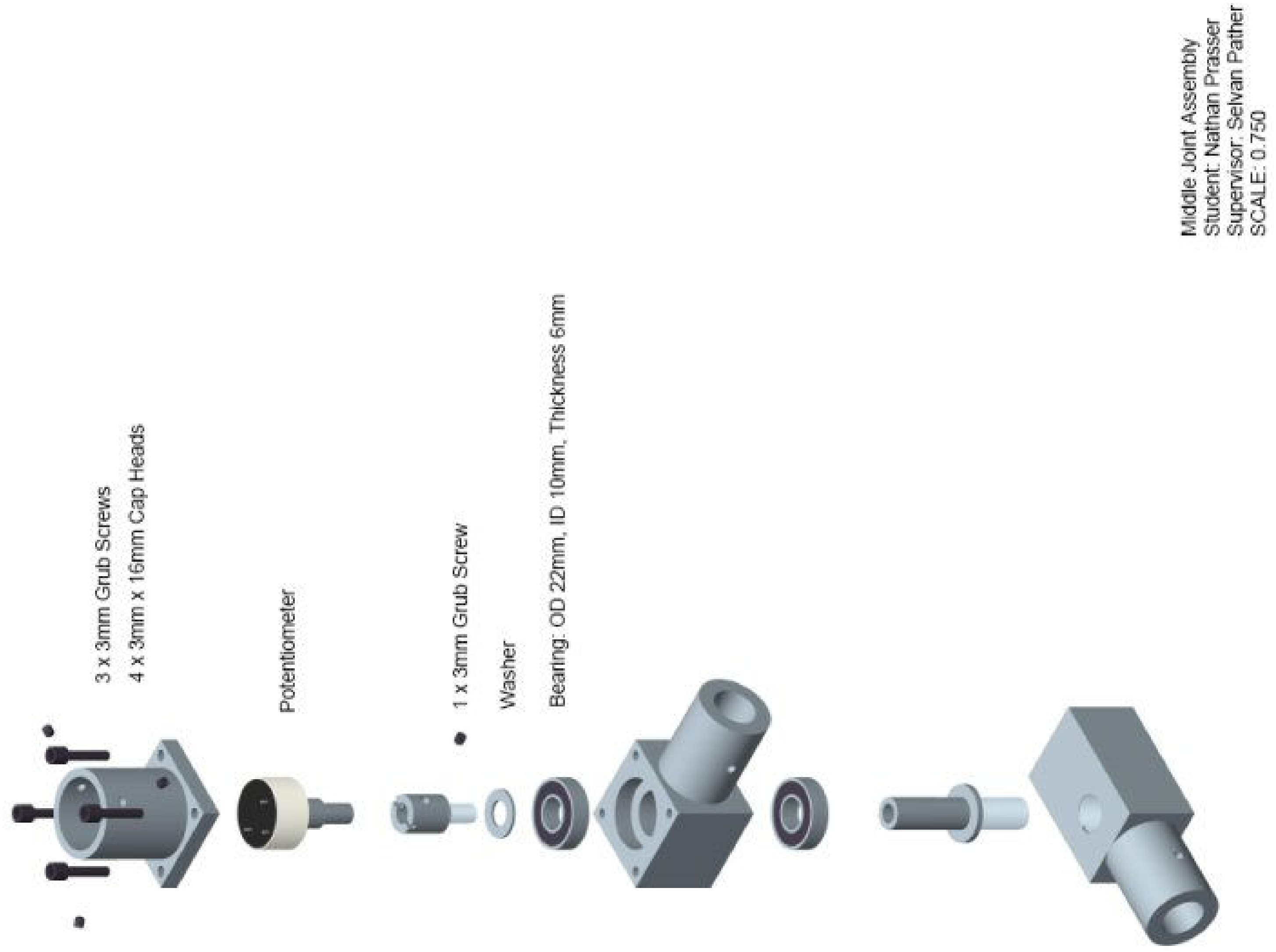

Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 1,500
24
14
3
M10 drilled and tapped
20
10
15
15
15
M15 drilled and tapped
30
6
7
30
46
Ø5
24

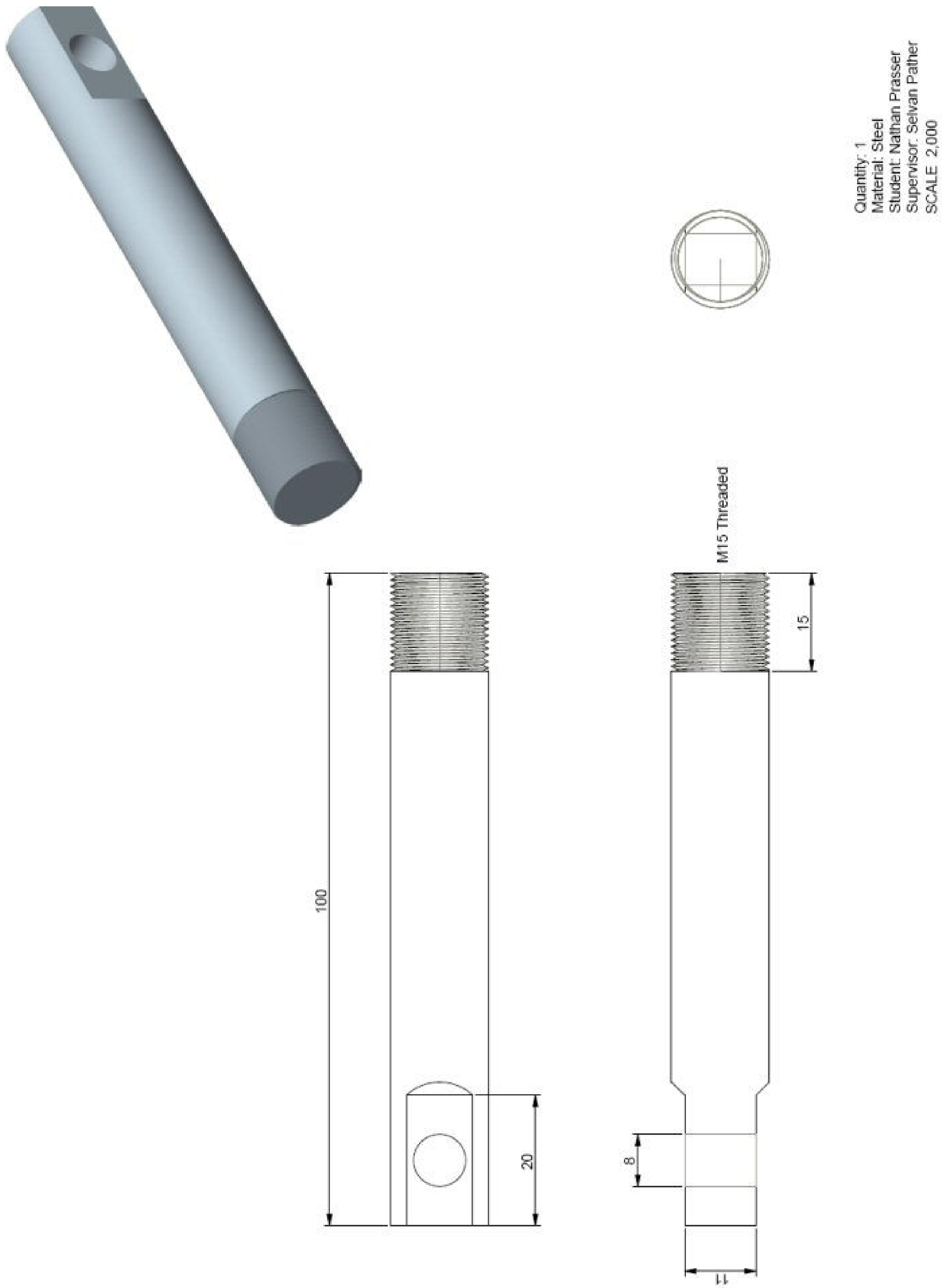

Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 2,500
M10 Threaded
10
20
M6 Drilled and Tapped
Ø18
18
15
27
2
3

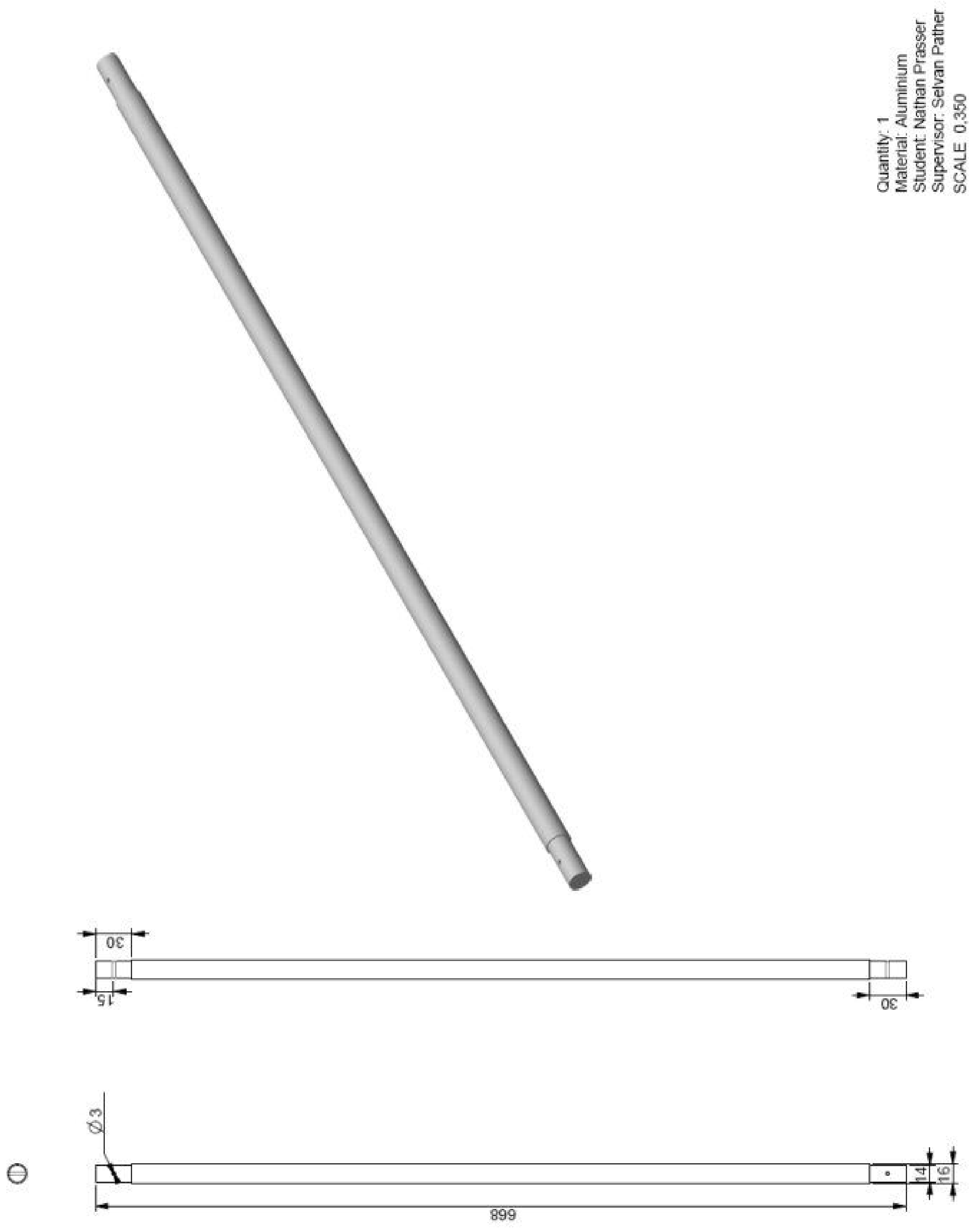

Quantity: 2
Material: Steel
Student: Nathan Prasser
Supervisor: Selvan Pather
Scale: 3.000
M3 Drilled and Tapped
M6 Threaded
12
6,3
6.3 mm drill hole should be Potentiometer shaft fit
Ø 12
Ø 6,3
SCALE 3,000
5,5
1,5
2
11
14
12

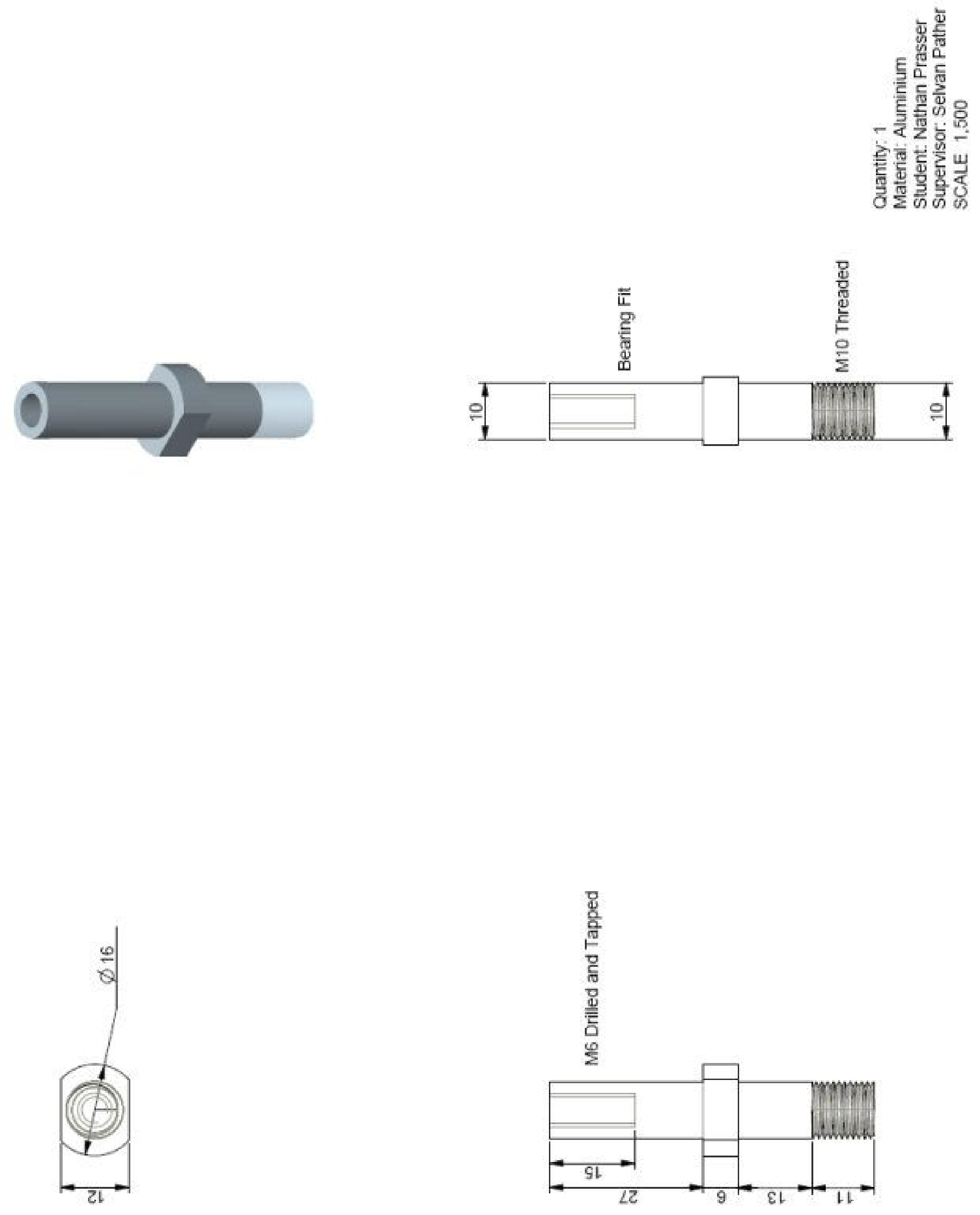

SCALE 2,500
Quantity: 2
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 3,000
28
24
14
30
3
32
22
14
3
3,5
3,5
4 x M3 drilled and tapped
32
6
11

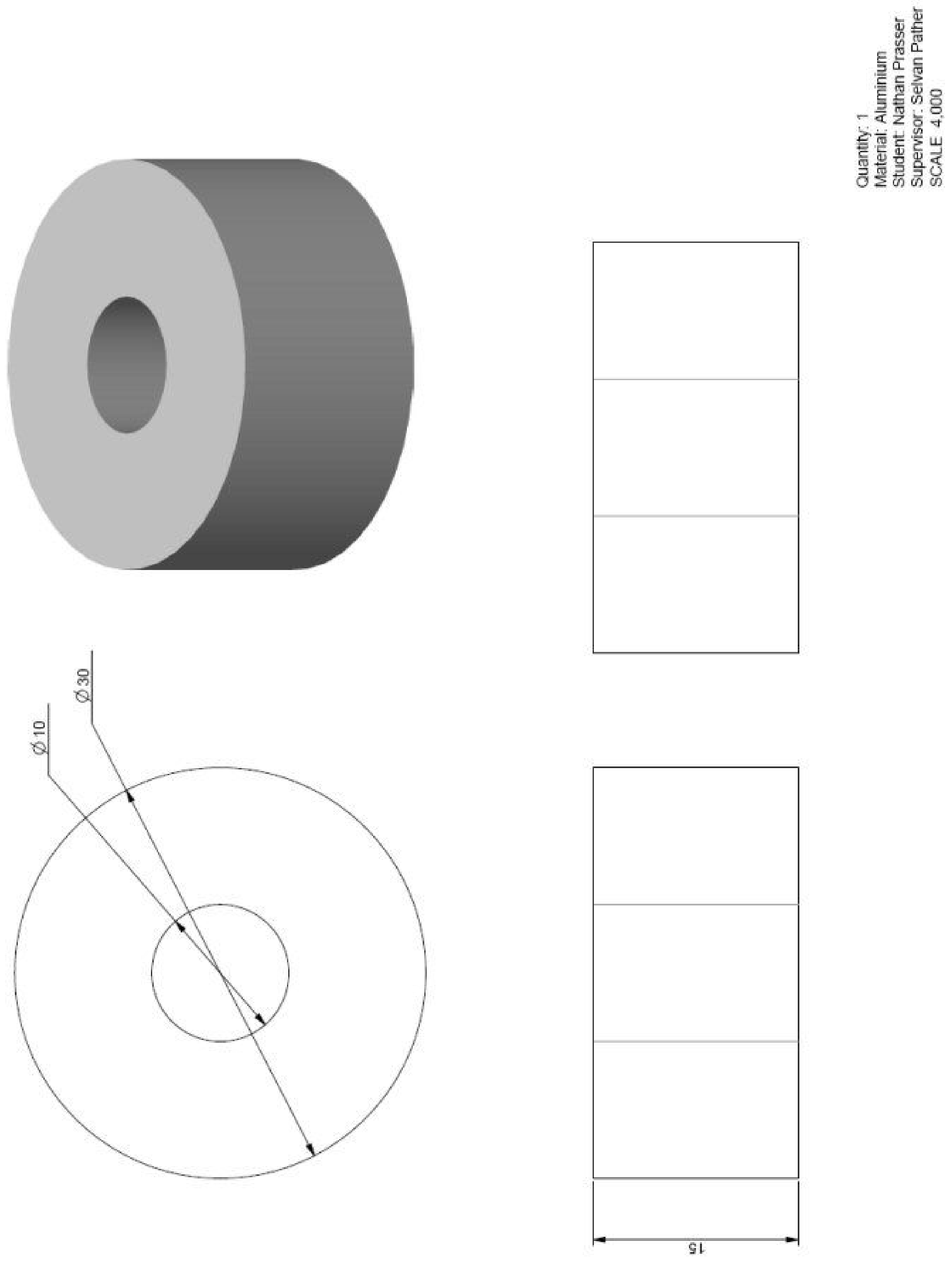

Quantity: 2
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 3,000
9
30
6
32
32
3,5
3,5
Ø3,5
36
28
22,5
Drill and Tap 3 x M3
8,5


Middle Joint Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE: 0.750
3 x 3mm Grub Screws
4 x 3mm x 16mm Cap Heads
Potentiometer
1 x 3mm Grub Screw
Washer
Bearing: OD 22mm, ID 10mm, Thickness 6mm

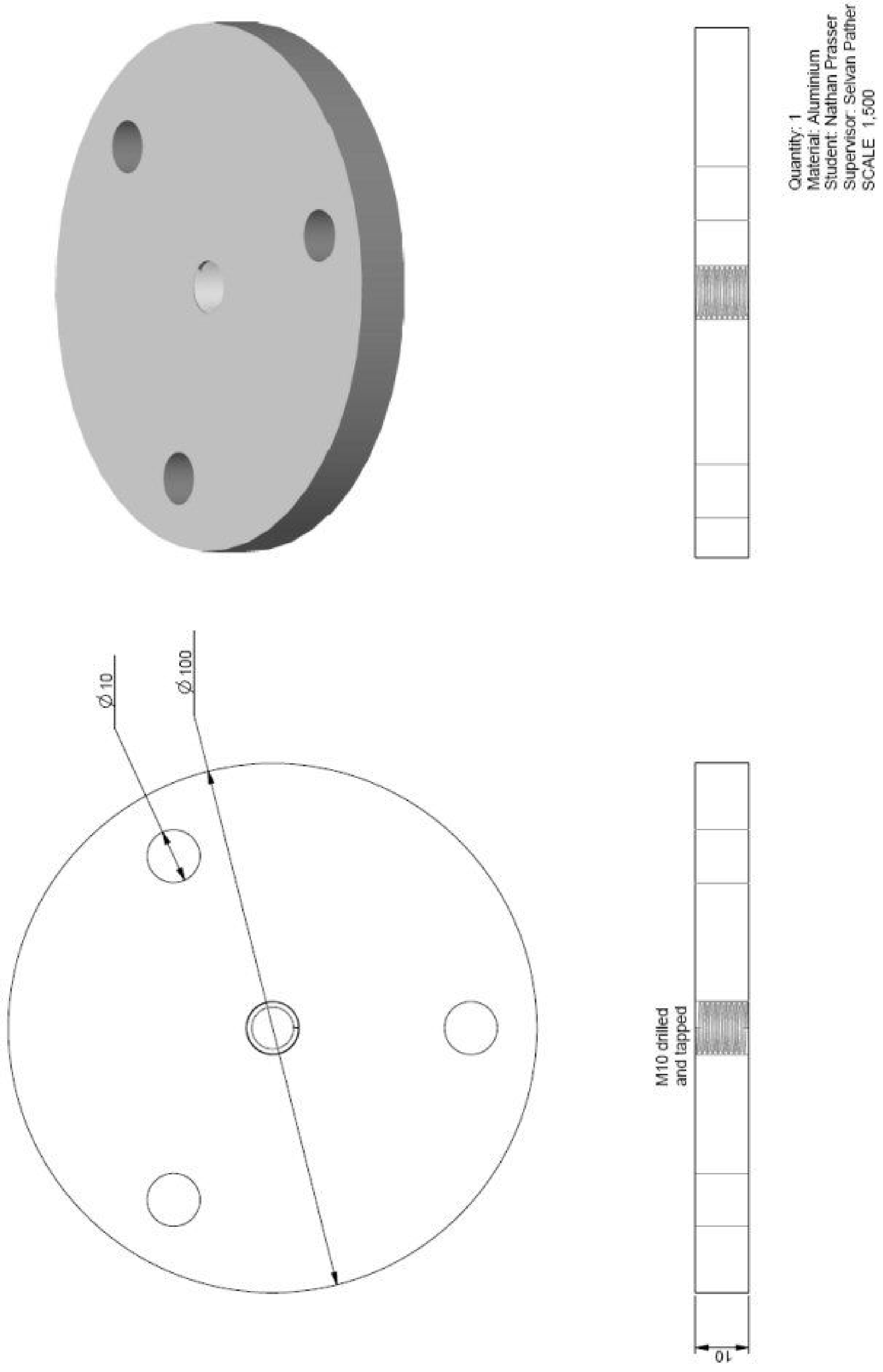

Quantity: 1
Material: Steel
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 2,000
M15 Threaded
15
100
20
8
11

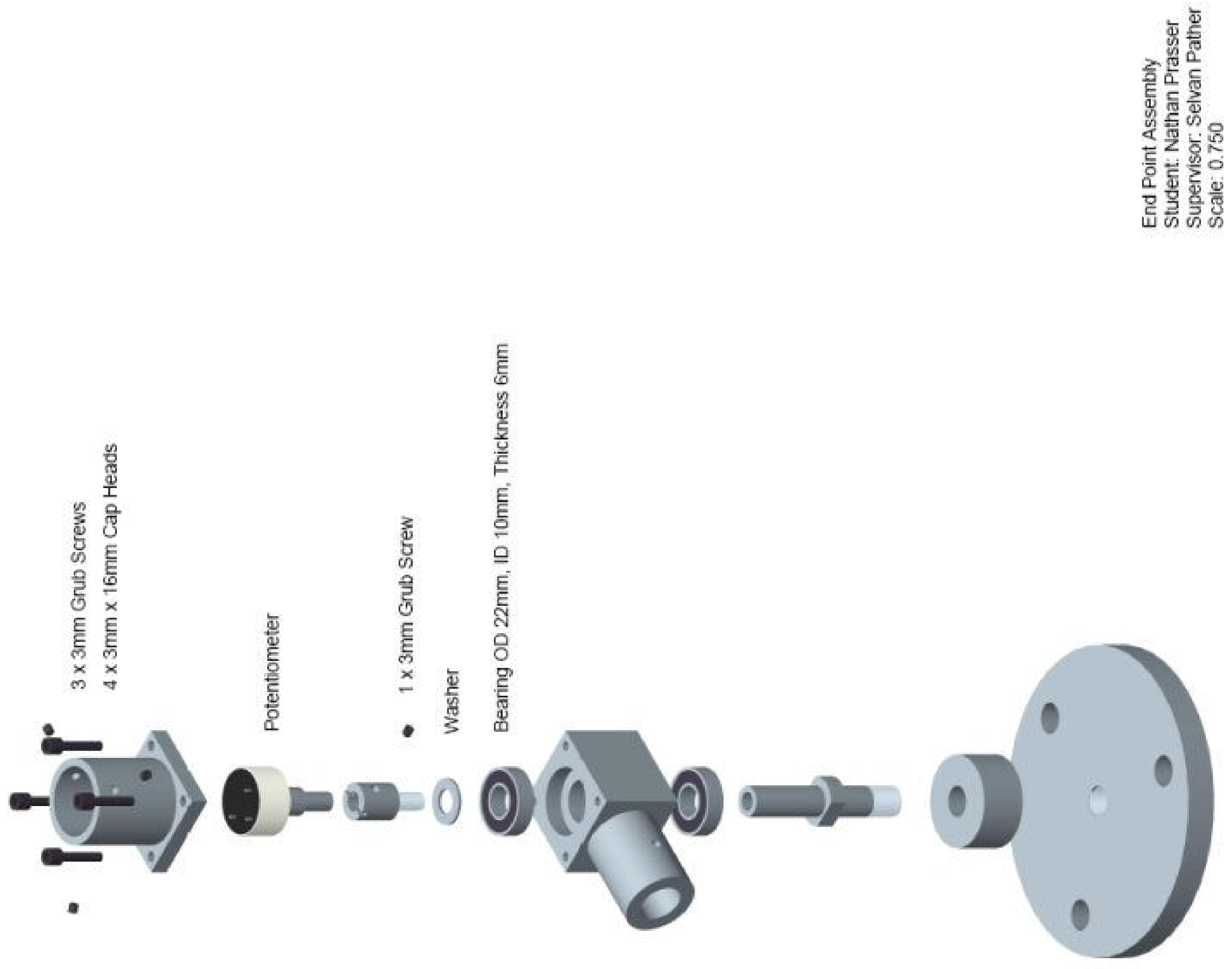

Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 0,350
30
15
30
Ø3
899
14
16

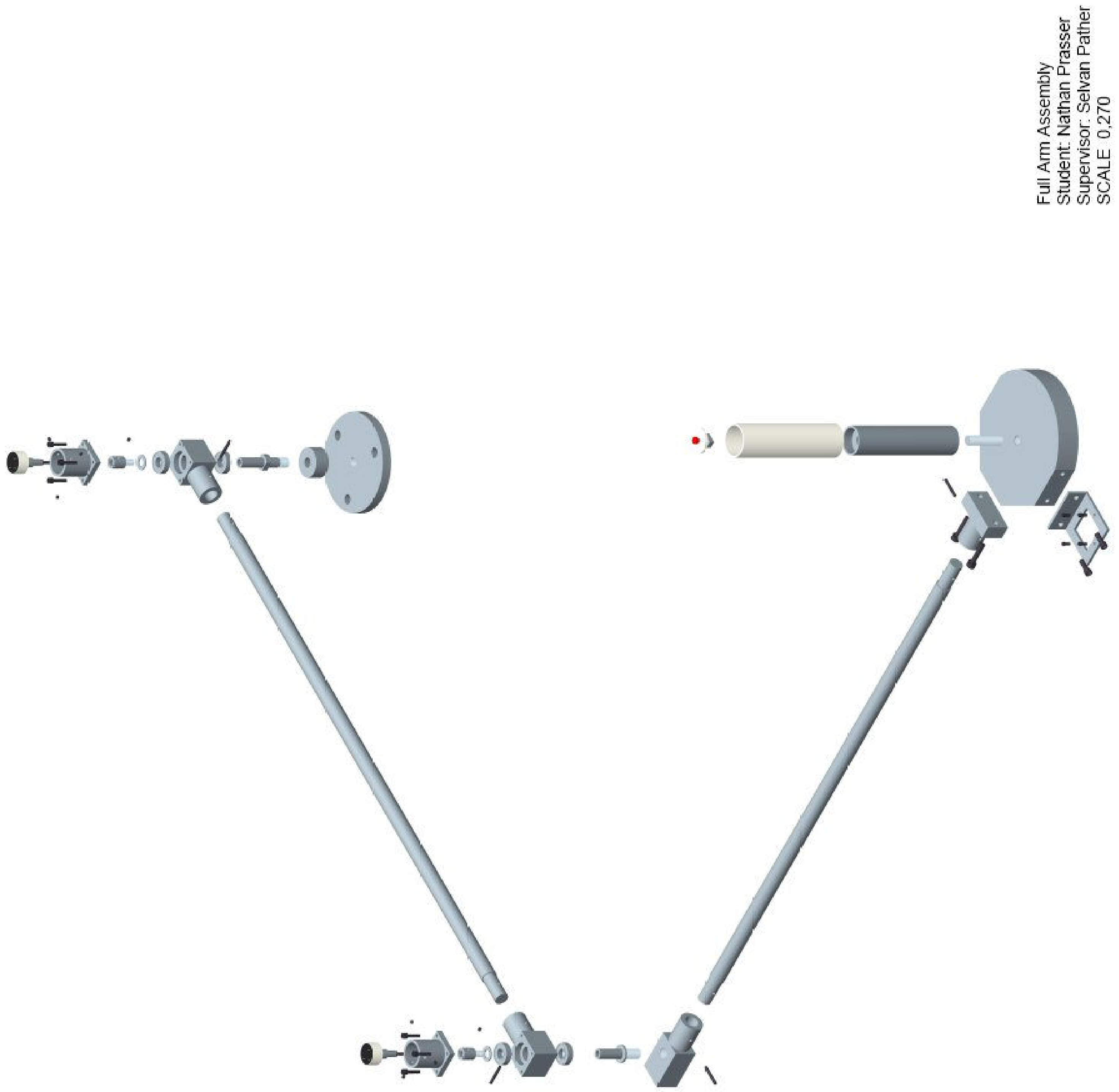

Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 1,500
Bearing Fit
M10 Threaded
10
10
M6 Drilled and Tapped
Ø 16
12
15
27
6
13
11

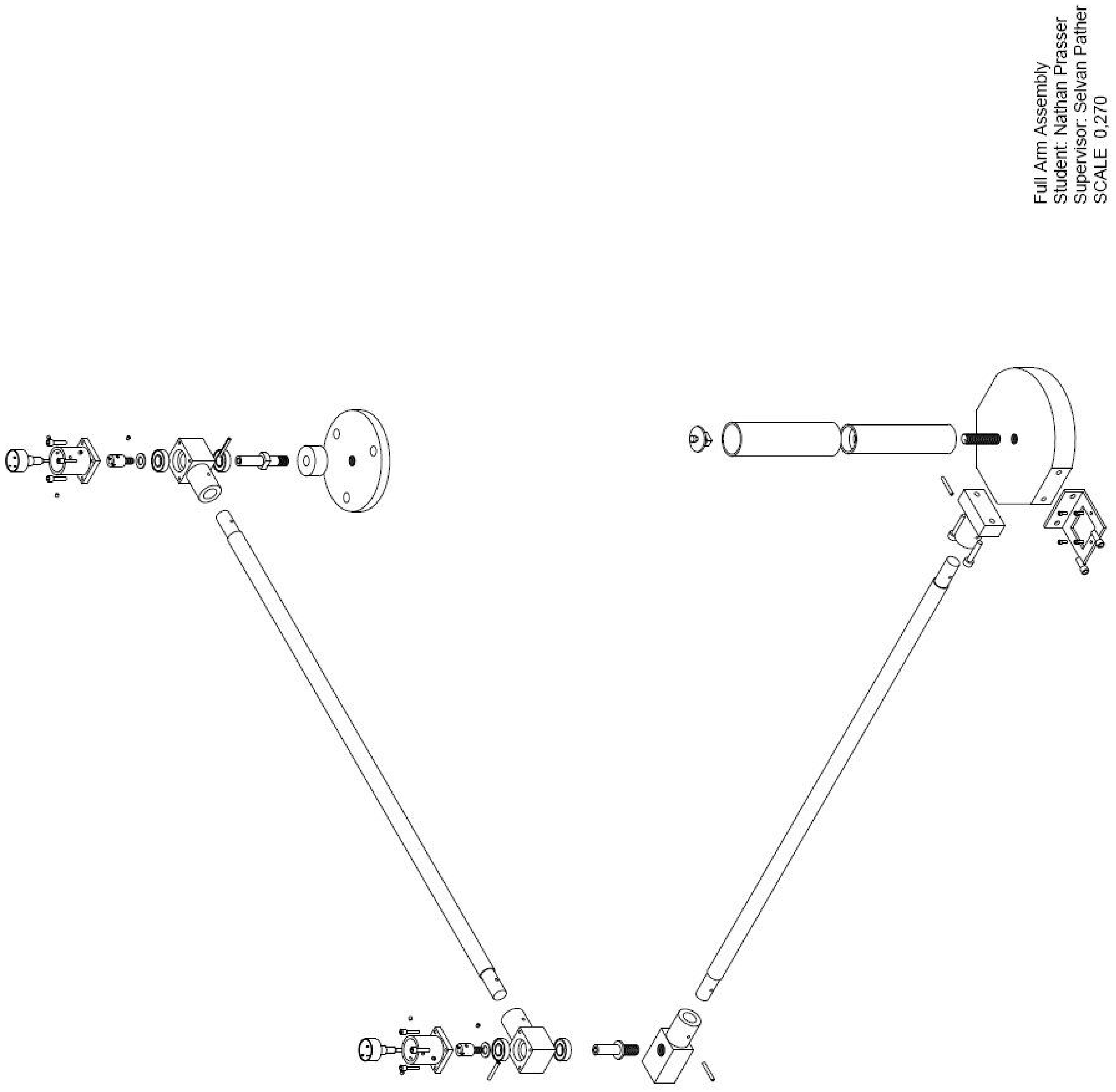


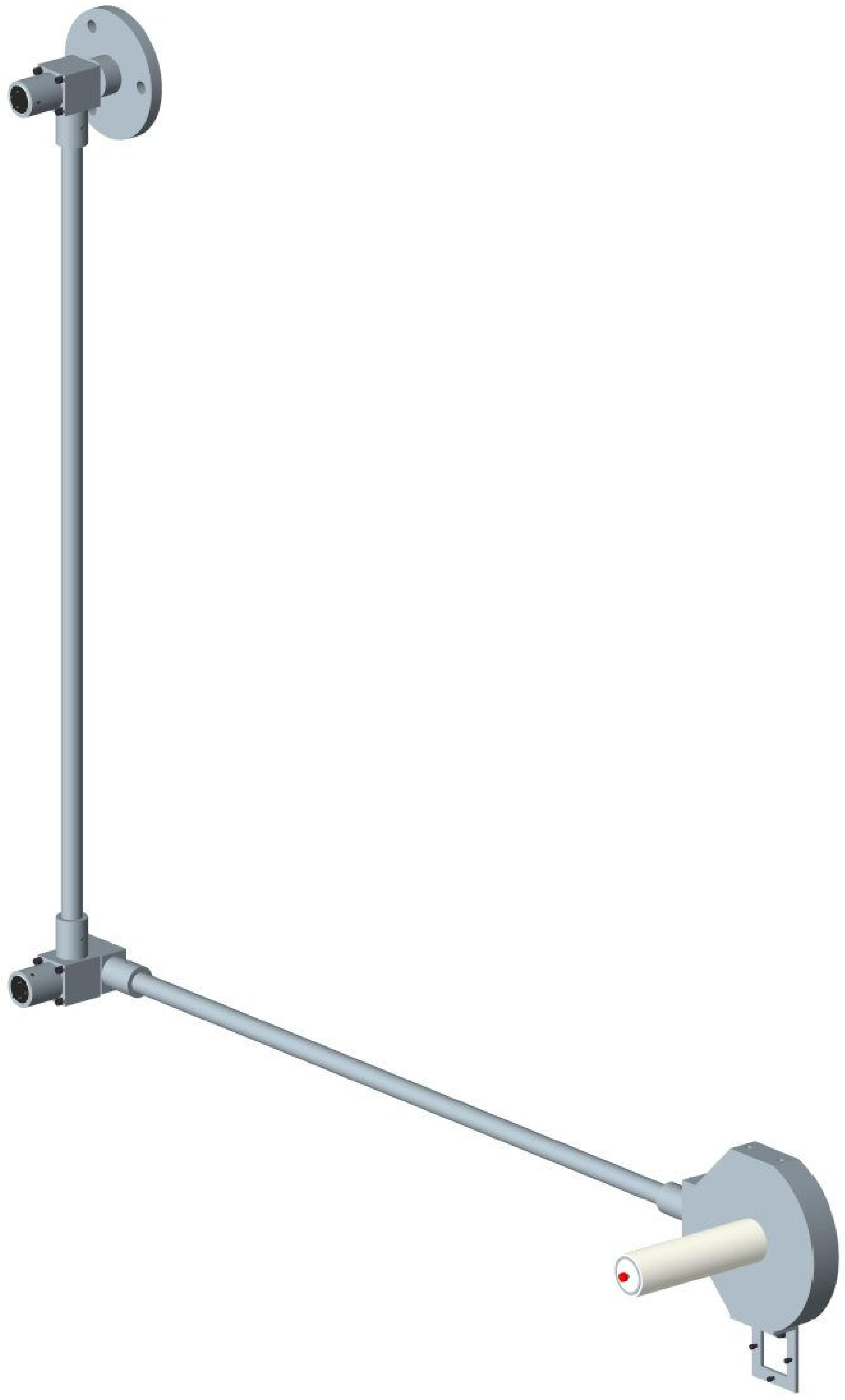
Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 4,000


Quantity: 1
Material: Aluminium
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 1,500
Ø 10
Ø 100
M10 drilled and tapped
10


3 x 3mm Grub Screws
4 x 3mm x 16mm Cap Heads
Potentiometer
1 x 3mm Grub Screw
Washer
Bearing OD 22mm, ID 10mm, Thickness 6mm
End Point Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
Scale: 0.750


Full Arm Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 0,270


Full Arm Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE 0,270



Full Assembly
Student: Nathan Prasser
Supervisor: Selvan Pather
SCALE: 0.300

Appendix C: Programming Code

Section 1: GUI Code

SplashScreen.vb

```
Imports System.io
Public NotInheritable Class SplashScreen

    Private Sub SplashScreen1_Load(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles Me.Load

        'If directory exists do nothing otherwise create directory
        If Directory.Exists("C:\Project\Tracking\" & DateString) = False Then
            Directory.CreateDirectory("C:\Project\Tracking\" & DateString)
        End If

        'Enable Timer1
        Timer1.Enabled = True

        'Application info
        If My.Application.Info.Title <> "" Then
            ApplicationTitle.Text = My.Application.Info.Title
        Else
            'If the application title is missing, use the application name, without
            'the(extension)
            ApplicationTitle.Text = System.IO.Path.GetFileNameWithoutExtension _
            (My.Application.Info.AssemblyName)
        End If
        'version info
        Version.Text = System.String.Format(Version.Text, _
        My.Application.Info.Version.Major, My.Application.Info.Version.Minor)
    End Sub

    Private Sub Timer1_Tick(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles Timer1.Tick

        'Starts RS232COM routine and hides it in the background
        RS232COM.Show()
        RS232COM.Hide()

        'Show calibration window
        Calibration.Show()

        'Hide the splashscreen
        Me.Hide()

        'Disable timer1
        Timer1.Enabled = False
    End Sub
End Class
```

Calibration.vb

```
Public Class Calibration

    Private Sub Calibration_Load(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles Me.Load

        'Enable timer1
        Timer1.Enabled = True
    End Sub

    Private Sub Configuration_Paint(ByVal sender As Object, ByVal e As _
    System.Windows.Forms.PaintEventArgs) Handles Me.Paint

        'Write the following text into the calibration window
        Dim heading As New Font("Arial", 18, FontStyle.Bold Or FontStyle.Underline)
        Dim subheading As New Font("arial", 20, FontStyle.Bold)
        Dim gtext As New Font("arial", 12, FontStyle.Regular)
        e.Graphics.DrawString("Rehabilitation Technologies", heading, Brushes.Black, 30, 10)
        e.Graphics.DrawString("IMPORTANT", subheading, Brushes.Red, 110, 60)
        e.Graphics.DrawString("Please position the crosshairs above the cross", gtext, _
        Brushes.Black, 20, 100)
        e.Graphics.DrawString("and HOLD the red button located on the top of", gtext, _
        Brushes.Black, 20, 120)
        e.Graphics.DrawString("the handle until this window disappears.", gtext, _
        Brushes.Black, 20, 140)
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) _
    Handles Timer1.Tick

        'Timer2 enabled
        Timer2.Enabled = True
        If RS232COM.ButtonControl.Text = "Button Pushed" Then

            'Clear buttontext to disable button
            RS232COM.ButtonControl.Clear()

            'Show application window, hide calibration window
            ApplicationWindow.Show()
            Me.Hide()
        End If
    End Sub

    Private Sub EndPOT_TextChanged(ByVal sender As System.Object, ByVal e As _
    System.EventArgs) Handles EndPOT.TextChanged

        'If textchanges enable timer1
        Timer1.Enabled = True
    End Sub
End Class
```

RS232COM.vb

```
Public Class RS232COM
    Dim WithEvents serialPort As New IO.Ports.SerialPort

    Private Sub RS232(ByVal sender As System.Object, ByVal e As System.EventArgs) _
    Handles MyBase.Load

        'If serial port is active close it so transfer protocols can be established
        If serialPort.IsOpen Then
            serialPort.Close()
        End If

        'Define transfer protocols
        Try
            serialPort.PortName = "COM1"
            serialPort.BaudRate = 9600
            serialPort.Parity = IO.Ports.Parity.None
            serialPort.DataBits = 8
            serialPort.StopBits = IO.Ports.StopBits.One
            serialPort.Open()
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try

    End Sub

    Private Sub DataReceived(ByVal sender As Object, ByVal e As _
    System.IO.Ports.SerialDataReceivedEventArgs) Handles serialPort.DataReceived

        'Get packet of data from COM1 port
        ReceiveData.Invoke(New myDelegate(AddressOf updateTextBox), New Object() {})
    End Sub

    Public Delegate Sub myDelegate()
    Dim length As Integer = 0
    Public Sub updateTextBox()
        ReceiveData.Font = New Font("Garamond", 12.0!, FontStyle.Bold)

        'hold data from previous calculation
        ResultCopyPrev.Text = ResultCopy.Text

        'copy current data
        ResultCopy.Text = ReceiveData.Text

        'clear reveivedata to differentiate between different rs232 communications
        ReceiveData.Clear()

        'middle potentiometer is sorted and converted to an array
        If ResultCopyPrev.Text = "6" Then
            If ResultCopy.Text.Length < 8 Then
                MiddlePOT.Text = ResultCopy.Text
                MiddlePOT.Text.ToCharArray()
                'MiddlePOT.Text = CDbl(MiddlePOT.Text)
                ApplicationWindow.ControlX.Text = MiddlePOT.Text
            End If
        End If

        'end potentiometer is sorted and converted to an array
        If ResultCopyPrev.Text = "7" Then
            If ResultCopy.Text.Length < 8 Then
                EndPOT.Text = ResultCopy.Text
                EndPOT.Text.ToCharArray()
                'EndPOT.Text = CDbl(EndPOT.Text)
                ApplicationWindow.ControlY.Text = EndPOT.Text
            End If
        End If

        'Initial button push for calibration
        If ResultCopy.Text = "b" Then
            ButtonControl.Text = "Button Pushed"
        End If

        'Character e can be received at anytime after calibration, this toggles
        'the cursor between the mouse and ARM
        If ResultCopy.Text = "e" Then
            If ButtonControl.Text = Nothing Then
                ButtonControl.Text = "Button Pushed"
            Else
                ButtonControl.Text = ""
            End If

        End If

        'Get next data set
        ReceiveData.AppendText(serialPort.ReadExisting)
        ReceiveData.ScrollToCaret()
    End Sub
End Class
```

ApplicationWindow.vb

```
Imports System.IO
Imports System.math
Public Class ApplicationWindow

    'initiate n counter, n counter is for unique file creation
    Dim n As Integer = 1

    'INITIAL WINDOW LOAD ITEMS

    Public Sub ApplicationWindow_Load(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles MyBase.Load

        'Load window state set to maximized
        Me.WindowState = FormWindowState.Maximized

        ' Create unique session logging
        Dim m As Integer = 1
        While Directory.Exists(My.Application.Info.DirectoryPath & "\Tracking\" _
        & DateString & "\Session " & m & " Easy") = True Or _
        Directory.Exists(My.Application.Info.DirectoryPath & "\Tracking\" _
        & DateString & "\Session " & m & " Medium") = True Or Directory.Exists _
        (My.Application.Info.DirectoryPath & "\Tracking\" & DateString _
        & "\Session " & m & " Hard")
            m = m + 1
        End While
        n = m

        'Close the calibration window
        Calibration.Close()

        'Set ARM to be the inital startup contorl
        If RS232COM.ButtonControl.Text = "" Then
            RS232COM.ButtonControl.Text = "Button Pushed"
        End If

        'Reposition Arm Ative sign to suit window size
        ArmActive.Location = New Point(Width - 80, 7)

        'Ensure that the control textboxs are empty
        ControlX.Clear()
        ControlY.Clear()

        'Initiate position control subroutines
        Timer1.Enabled = True

    End Sub


    ' TOOL STRIP ITEMS

    ' Difficulty items

    ' TSM = Tool strip menu item
```

```
Private Sub EasyTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles EasyMenuItem.Click

    'If Textbox1 = 1 (high) then the user has just returned from reviewing past
    'sessions and has changed the difficulty without entering the base area thus
    'creating new points/a new file, without this step the unique file
    'incrementation would be incremented twice, then when the program is restarted
    'files would be over written thus putting sessions out of order. the textbox
    'is set to low (texbox1 = "") when the mouse is moved within the base area
    If TextBox1.Text <> "1" Then
        n = n + 1
    End If

    'List boxs have to be cleared if difficulty changes so that information
    'from previous session does not carry onto next session
    Me.ListBox1.Items.Clear()
    Me.ListBox2.Items.Clear()

    'If easy is checked and the other two arn't
    Me.EasyMenuItem.Checked = True
    Me.MediumMenuItem.Checked = False
    Me.HardMenuItem.Checked = False

    'Hide all points apart from the first location of selected difficulty
    Me.Easy1.Location = New Point((Me.Size.Width + 150) - Me.Size.Width, _
    (Me.Size.Height + 125) - Me.Size.Height)
    Me.Easy1.Show()
    Me.Easy2.Hide()
    Me.Easy3.Hide()
    Me.Medium1.Hide()
    Me.Medium2.Hide()
    Me.Medium3.Hide()
    Me.Hard1.Hide()
    Me.Hard2.Hide()
    Me.Hard3.Hide()
End Sub

Private Sub MediumTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles MediumMenuItem.Click

    'Everything is the same as above except difficulty medium
    'has been selected this time.
    If TextBox1.Text <> "1" Then
        n = n + 1
    End If
    Me.ListBox1.Items.Clear()
    Me.ListBox2.Items.Clear()
    Me.EasyMenuItem.Checked = False
    Me.MediumMenuItem.Checked = True
    Me.HardMenuItem.Checked = False
    Me.Easy1.Hide()
    Me.Easy2.Hide()
    Me.Easy3.Hide()
    Me.Medium1.Location = New Point(200, (Me.Size.Height / 2))
    Me.Medium1.Show()
    Me.Medium2.Hide()
    Me.Medium3.Hide()
    Me.Hard1.Hide()
    Me.Hard2.Hide()
    Me.Hard3.Hide()
End Sub

Private Sub HardTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles HardMenuItem.Click

    'Hard difficulty selected this time
    If TextBox1.Text <> "1" Then
        n = n + 1
    End If
    Me.ListBox1.Items.Clear()
    Me.ListBox2.Items.Clear()
    Me.EasyMenuItem.Checked = False
    Me.MediumMenuItem.Checked = False
    Me.HardMenuItem.Checked = True
    Me.Easy1.Hide()
    Me.Easy2.Hide()
    Me.Easy3.Hide()
    Me.Medium1.Hide()
    Me.Medium2.Hide()
    Me.Medium3.Hide()
    Me.Hard1.Location = New Point(100, Me.Size.Height - 100)
    Me.Hard1.Show()
    Me.Hard2.Hide()
    Me.Hard3.Hide()
End Sub

'Use random points or test points

Private Sub TestLocationsToolStripMenuItem_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles TestLocationsToolStripMenuItem.Click
    Me.TestLocationsToolStripMenuItem.Checked = True
    Me.RandomLocationsToolStripMenuItem.Checked = False
End Sub

Private Sub RandomLocationsToolStripMenuItem_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles RandomLocationsToolStripMenuItem.Click
    Me.TestLocationsToolStripMenuItem.Checked = False
    Me.RandomLocationsToolStripMenuItem.Checked = True
End Sub

'Changes the movement sensitivity of the arm

Private Sub LowSensitivity_Click(ByVal sender As System.Object, ByVal e As _
System.EventArgs) Handles LowSensitivity.Click
    Me.LowSensitivity.Checked = True
    Me.MediumSensitivity.Checked = False
    Me.HighSensitivity.Checked = False
End Sub

Private Sub MediumSensitivity_Click(ByVal sender As System.Object, ByVal e As _
System.EventArgs) Handles MediumSensitivity.Click
    Me.LowSensitivity.Checked = False
    Me.MediumSensitivity.Checked = True
    Me.HighSensitivity.Checked = False
```

```
End Sub

Private Sub HighSensitivity_Click(ByVal sender As System.Object, ByVal e As _
System.EventArgs) Handles HighSensitivity.Click
    Me.LowSensitivity.Checked = False
    Me.MediumSensitivity.Checked = False
    Me.HighSensitivity.Checked = True
End Sub

'Grid enable/disable options
Private Sub OnToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e _
As System.EventArgs) Handles OnToolStripMenuItem.Click
    OnToolStripMenuItem.Checked = True
    OffToolStripMenuItem.Checked = False
    MyBase.Refresh()
End Sub

Private Sub OffToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e _
As System.EventArgs) Handles OffToolStripMenuItem.Click
    OnToolStripMenuItem.Checked = False
    OffToolStripMenuItem.Checked = True
    MyBase.Refresh()
End Sub


'Review Past exercises completed

Public Sub ReviewTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles ReviewToolStripMenuItem.Click

    'Increment unique files creation.
    If Me.TestLocationsToolStripMenuItem.Checked = True Then
        n = n + 1
    End If

    'Clear all previous data locations
    Me.ListBox1.Items.Clear()
    Me.ListBox2.Items.Clear()

    'Hide the application window and show the review window
    Me.Hide()
    Review.Show()
End Sub

'Help menu, helps users understand how to use the program

Private Sub HelpTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles HelpToolStripMenuItem.Click
    Help.Show()
End Sub

'Lists Version, creation date, users

Private Sub AboutTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles AboutToolStripMenuItem.Click
    About.Show()
End Sub
```

```
'Quits program properly, currently close at top corner does not close
'application properly
Private Sub QuitTSM(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles QuitToolStripMenuItem.Click
    SplashScreen.Close()
    Me.Close()
    Review.Close()
    RS232COM.Close()
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' PICTURE BOXS - POINTS TO TOUCH MOVE

'Easy
Private Sub Easy1_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Easy1.MouseHover

    'If the test locations is selected, on mouse hover of image 1 of easy
    'hide image 1 of easy and show image 2
    If Me.TestLocationsToolStripMenuItem.Checked = True Then
        Me.Easy1.Hide()
        Me.Easy2.Show()
        MyBase.Refresh()

        'Position image relative to window size
        Me.Easy2.Location = New Point(Me.Size.Width - 200, (125 + Me.Size.Height) _
        - Me.Size.Height)

        'If random locations is selected place image one of easy in random locations
        'anywhere onscreen
    ElseIf Me.RandomLocationsToolStripMenuItem.Checked = True Then
        Dim rand As New Random()
        Dim xlocation As Integer
        Dim ylocation As Integer
        xlocation = rand.Next(50, Me.Width)
        ylocation = rand.Next(50, Me.Height)
        Me.Easy1.Location = New Point(xlocation - 50, ylocation - 50)
    End If
End Sub

'Test locations has been selected continue hiding and showing image
'to give the illusion that the image is moving
Private Sub Easy2_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Easy2.MouseHover
    Me.Easy2.Hide()
    Me.Easy3.Show()
    MyBase.Refresh()
    Me.Easy3.Location = New Point((Me.Size.Width / 2 - 25), (Me.Size.Height - 125))
End Sub
Private Sub Easy3_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Easy3.MouseHover
    Me.Easy3.Hide()
    Me.Easy1.Show()
    MyBase.Refresh()
    Me.Easy1.Location = New Point((Me.Size.Width + 150) - Me.Size.Width, _
    (Me.Size.Height + 125) - Me.Size.Height)
```

```
End Sub

'Medium

'The following six subroutines are the same as the previous three
'except handling medium difficulty and hard difficulty
Private Sub Medium1_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Medium1.MouseHover
    If Me.TestLocationsToolStripMenuItem.Checked = True Then
        Me.Medium1.Hide()
        Me.Medium2.Show()
        MyBase.Refresh()
        Me.Medium2.Location = New Point(Me.Size.Width - 250, 150)
    ElseIf Me.RandomLocationsToolStripMenuItem.Checked = True Then

        Dim rand As New Random()
        Dim xlocation As Integer
        Dim ylocation As Integer
        xlocation = rand.Next(30, Me.Width)
        ylocation = rand.Next(30, Me.Height)
        Me.Medium1.Location = New Point(xlocation - 30, ylocation - 30)
    End If
End Sub

Private Sub Medium2_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Medium2.MouseHover
    Me.Medium2.Hide()
    Me.Medium3.Show()
    MyBase.Refresh()
    Me.Medium3.Location = New Point((Me.Size.Width - 250), (Me.Size.Height - 150))
End Sub

Private Sub Medium3_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Medium3.MouseHover
    Me.Medium3.Hide()
    Me.Medium1.Show()
    MyBase.Refresh()
    Me.Medium1.Location = New Point(200, (Me.Size.Height / 2))
End Sub

'Repeated for Hard
Private Sub Hard1_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Hard1.MouseHover
    If Me.TestLocationsToolStripMenuItem.Checked = True Then
        Me.Hard1.Hide()
        Me.Hard2.Show()
        MyBase.Refresh()
        Me.Hard2.Location = New Point(Me.Size.Width / 2, 100)
    ElseIf Me.RandomLocationsToolStripMenuItem.Checked = True Then

        Dim rand As New Random()
        Dim xlocation As Integer
        Dim ylocation As Integer
        xlocation = rand.Next(15, Me.Width)
        ylocation = rand.Next(15, Me.Height)
        Me.Hard1.Location = New Point(xlocation - 15, ylocation - 15)
    End If
End Sub

Private Sub Hard2_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Hard2.MouseHover
    Me.Hard2.Hide()
    Me.Hard3.Show()
    MyBase.Refresh()
    Me.Hard3.Location = New Point(Me.Size.Width - 100, Me.Size.Height - 100)
End Sub

Private Sub Hard3_MouseHover(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles Hard3.MouseHover
    Me.Hard3.Hide()
    Me.Hard1.Show()
    MyBase.Refresh()
    Me.Hard1.Location = New Point(100, Me.Size.Height - 100)
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'WRITE COORDINATES OF MOUSEMOVEMENTS TO UNIQUE LOCATIONS, X AND Y LOCATIONS SAVED
'TO SINGULAR FILES

Dim x As IO.StreamWriter
Dim y As IO.StreamWriter

Private Sub Form1_MouseMove(ByVal sender As Object, ByVal e As _
System.Windows.Forms.MouseEventArgs) Handles Me.MouseMove

    'This textbox is used to allow unique file incrementation, clear textbox1
    'to show that the user has created a unique file, thus avoiding file
    'overwritting when program is restarted
    TextBox1.Clear()
    If Me.TestLocationsToolStripMenuItem.Checked = True Then

        'Clear all precious values stored in the listboxs
        ListBox1.Hide()
        ListBox2.Hide()

        'Generate random numer for x and y coordinate
        Dim locationtextx As String = String.Format("{0}", e.X)
        Dim locationtexty As String = String.Format("{0}", e.Y)

        'Put these random number into listboxs
        ListBox1.Items.Insert(0, locationtextx)
        ListBox2.Items.Insert(0, locationtexty)
        Dim i As Integer

        'Creates unique directorys and stores the coordiates that the cursor
        'has moved. The x and y coordinates are also stored as seperate files
        If Me.EasyMenuItem.Checked = True Then
            Directory.CreateDirectory(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Easy")
        End If
        If Me.MediumMenuItem.Checked = True Then
            Directory.CreateDirectory(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Medium")
        End If
        If Me.HardMenuItem.Checked = True Then
            Directory.CreateDirectory(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Hard")
        End If
        If Me.EasyMenuItem.Checked = True Then
            x = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Easy\XCoordinate.txt")
            y = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Easy\YCoordinate.txt")
        End If
        If Me.MediumMenuItem.Checked = True Then
            x = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Medium\XCoordinate.txt")
            y = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Medium\YCoordinate.txt")
        End If
        If Me.HardMenuItem.Checked = True Then
            x = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Hard\XCoordinate.txt")
            y = New IO.StreamWriter(My.Application.Info.DirectoryPath & "\Tracking\" _
            & DateString & "\Session " & n & " Hard\YCoordinate.txt")
        End If

        'Counts number of values in listboxs and adds string "EOF" to the very
        'end, this allows for the end of file to be easily found when reading
        'values during review.vb
        For i = 0 To ListBox1.Items.Count - 1
            x.WriteLine(ListBox1.Items.Item(i))
        Next
        x.WriteLine("EOF")
        x.Close()
        For i = 0 To ListBox2.Items.Count - 1
            y.WriteLine(ListBox2.Items.Item(i))
        Next
        y.WriteLine("EOF")
        y.Close()
    End If
End Sub

'Draw grid on working area
Private Sub ApplicationWindow_Paint(ByVal sender As Object, ByVal e As _
System.Windows.Forms.PaintEventArgs) Handles Me.Paint
    If Me.OnToolStripMenuItem.Checked = True Then

        Dim xboxes As Integer = Width / 12

        '50 is to correct for the toolbar and start menu bar
        Dim yboxes As Integer = (Height - 50) / 9

        'remove tool bar pixels
        Dim yplot As Integer = 25
        Dim xplot As Integer
        Dim counter As Integer = 0

        'For an even set of spaces draw horizontal lines to display a grid pattern
        For counter = 0 To 9
```

```
            yplot = yplot + yboxes
            Dim pts() As Point = {New Point(0, yplot), New Point(Width, yplot)}

            e.Graphics.SmoothingMode = Drawing2D.SmoothingMode.AntiAlias

            e.Graphics.DrawCurve(Pens.Black, pts)
        Next counter

        'For an even set of spaces draw vertical lines to display a grid pattern
        For counter = 0 To 12

            xplot = xplot + xboxes
            Dim pts() As Point = {New Point(xplot, 0), New Point(xplot, Height)}

            e.Graphics.SmoothingMode = Drawing2D.SmoothingMode.AntiAlias

            e.Graphics.DrawCurve(Pens.Black, pts)
        Next counter
    End If

    'If grid turned off change base colour back to orignal colour
    If Me.OffToolStripMenuItem.Checked = True Then
        MyBase.BackColor = Color.Beige
    End If
End Sub

'If the window is resized this automatically repositions the images
Private Sub ApplicationWindow_Resize(ByVal sender As Object, ByVal e _
As System.EventArgs) Handles Me.Resize
    Me.Easy1.Location = New Point((Me.Size.Width + 150) - Me.Size.Width, _
    (Me.Size.Height + 125) - Me.Size.Height)
    Me.Easy2.Location = New Point(Me.Size.Width - 200, (125 + Me.Size.Height) _
    - Me.Size.Height)
    Me.Easy3.Location = New Point((Me.Size.Width / 2 - 25), _
    (Me.Size.Height - 125))
    Me.Medium1.Location = New Point(200, (Me.Size.Height / 2))
    Me.Medium2.Location = New Point(Me.Size.Width - 250, 150)
    Me.Medium3.Location = New Point((Me.Size.Width - 250), (Me.Size.Height - 150))
    Me.Hard1.Location = New Point(100, Me.Size.Height - 100)
    Me.Hard2.Location = New Point(Me.Size.Width / 2, 100)
    Me.Hard3.Location = New Point(Me.Size.Width - 100, Me.Size.Height - 100)
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'CURSOR CONTROL

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles Timer1.Tick
    Dim xposition As Double
    Dim yposition As Double
    Dim pi As Double = 3.14159265
    Dim midx As Double
    Dim midy As Double
    Dim midxdiff As Double
    Dim midydiff As Double
    Dim degreechange1 As Double
    Dim degreechange2 As Double
    Dim sensitivity As Double
    Dim armlength As Integer = 1500

    'Sets the sensitivity of the ARM
    If Me.LowSensitivity.Checked = True Then
        sensitivity = 300
    End If
    If Me.MediumSensitivity.Checked = True Then
        sensitivity = 200
    End If
    If Me.HighSensitivity.Checked = True Then
        sensitivity = 50
    End If

    'The routine would originally grab the end pot reading, perform x/y position
    'calculations and then read the middle pot and perform another x/y position
    'calculation. This resulted in a zigzag effect. Now both values are grabbed
    'and put into "control" textboxes, allowing the new values to be related to
    'eachother, and removing the zig zag effect. After each x/y calculation the
    'textboxes are cleared and the process repeats.
    If ControlX.Text <> Nothing Then
        If ControlY.Text <> Nothing Then

            'Adjusts the sesitivity. degreechange1 is inverted for calculation purposes
            degreechange1 = -RS232COM.EndPOT.Text / sensitivity
            degreechange2 = RS232COM.MiddlePOT.Text / sensitivity

            'Calculates x and y position
            xposition = (armlength * Cos(degreechange1)) + _
            (armlength * Sin(degreechange1 + degreechange2)) - 40
            yposition = (armlength * Sin(degreechange1)) + _
            (armlength * Cos(degreechange1 + degreechange2))

            'Positions cursor in the center of the screen by getting the screen size
            'and finding the error between its normal starting position and center
            'this value is then added to all plots. this calculation is only done once
            If MidXCalc.Text = Nothing Then
                midx = Me.Size.Width / 2
                midy = Me.Size.Height / 2
                midxdiff = armlength - midx - 20
                midydiff = armlength - midy
                MidXCalc.Text = midxdiff
                MidYCalc.Text = midydiff
            End If

            'If the toggle button has been pushed activate/deactivate ARM
            If RS232COM.ButtonControl.Text = "Button Pushed" Then

                'Allows the user to see when the arm has been activated
                ArmActive.Show()

                'Reposition cursor relative to the x/y position calculations
                System.Windows.Forms.Cursor.Position = New Point(xposition - _
                MidXCalc.Text, yposition - MidYCalc.Text)
            End If

            If RS232COM.ButtonControl.Text = "" Then
```

```
                'Allows the user to see when the ARM has been deactivated
                ArmActive.Hide()
            End If

            'Clears the control loop textboxes
            ControlX.Clear()
            ControlY.Clear()
        End If
    End If

    'Initiate Timer2
    Timer2.Enabled = True

End Sub

Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles Timer2.Tick
    Timer1.Enabled = True
End Sub

End Class
```

Review.vb

```
Imports System.IO

Public Class Review
    Dim n As Integer = 1

    ' LOAD REVIEW WINDOW INITAL ITEMS

    Private Sub Review_Load(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles Me.Load

        'Load window state relative to the main window stat
        Me.WindowState = ApplicationWindow.WindowState

        'Display helpful information in listboxes
        Me.DateBox.Text = "Choose Date"
        Me.SessionBox.Text = "Choose Session"

    End Sub

    ' TOOL STRIP ITEMS

    ' Return to the application window once finished reviewing

    Private Sub ReturnClick(ByVal sender As System.Object, ByVal e As System.EventArgs) _
    Handles ReturnToApplicationWindowToolStripMenuItem.Click

        'Counter for unique file creation
        If ApplicationWindow.TestLocationsToolStripMenuItem.Checked = True Then
            n = n + 1
        End If

        'This is used to produce unique file creation
        ApplicationWindow.TextBox1.Text = "1"

        'Show application window and hide review window
        ApplicationWindow.Show()
        Me.Close()
    End Sub

    ' Choose a date on which to review progress

    Private Sub DateBoxDropDown(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles DateBox.Click

        'Clear current state of listbox
        DateBox.Items.Clear()

        'Add all filenames to listbox and sort
        DateBox.Items.AddRange(IO.Directory.GetDirectories _
        (My.Application.Info.DirectoryPath & "\Tracking\"))
        DateBox.Sorted = True
    End Sub

    ' Choose a session related to date of which to review
```

```
Private Sub SessionBox_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles SessionBox.Click

    'If a date hasn't been choosen then do nothing
    If DateBox.Text = "Choose Date" Then
        SessionBox.Text = ("Choose Session")
    Else

        'Add all filenames to listbox
        SessionBox.Items.Clear()
        SessionBox.Items.AddRange(IO.Directory.GetDirectories(DateBox.Text))
    End If
End Sub


' DRAW PATH TAKEN

'This system reads the two files stored as the X and Y locations related to mouse
'movements and plots them.

Private Sub Form1_Paint(ByVal sender As Object, ByVal e As _
    System.Windows.Forms.PaintEventArgs) Handles _
    MyBase.Paint

    If SessionBox.Text = "Choose Session" Then

        Me.SessionBox.Text = "Choose Session"
    Else

        'Counts lines in x/y Coordinate text files so it can be implemented
        'into drawing code
        Dim count As New System.IO.StreamReader(Me.SessionBox.Text & "\XCoordinate.txt")
        Dim counter As Integer
        Dim EOF As String = ""
        Do Until EOF = "EOF"

            If count.ReadLine() <> "EOF" Then
                counter = counter + 1
            Else
                EOF = "EOF"
            End If
        Loop


        ' Below is code used for drawing lines, how it was done is
        ' x1/y1 are first loaded from text file, x2/y2 is second point from
        ' text file, it then draws a line between these two points.
        ' It then starts to loop again incrementing one point, the loop
        ' number is determined from the code just above which counts
        ' how many lines are in the file, this allows the loop to exit
        ' in optimal time

        Dim a As Integer
        Dim x As New System.IO.StreamReader(Me.SessionBox.Text & "\XCoordinate.txt")
        Dim y As New System.IO.StreamReader(Me.SessionBox.Text & "\YCoordinate.txt")
        Dim x1 As Integer
        Dim y1 As Integer
        Dim x2 As Integer
        Dim y2 As Integer

        x1 = x.ReadLine
        y1 = y.ReadLine
        x2 = x1
        y2 = y1

        For a = 1 To counter - 1

            x1 = x.ReadLine
            y1 = y.ReadLine
            Dim pts() As Point = {New Point(x1, y1), New Point(x2, y2)}

            e.Graphics.SmoothingMode = _
                Drawing2D.SmoothingMode.AntiAlias

            e.Graphics.DrawCurve(Pens.Black, pts)

            x2 = x1
            y2 = y1
        Next a
    End If
End Sub


Private Sub GOToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As _
System.EventArgs) Handles GOToolStripMenuItem.Click

    'Refresh to display plots and images
    MyBase.Refresh()

    'Determine from directory name if the file is easy, medium or hard
    'and display the correct images that apply to the difficulty
    If SessionBox.Text.EndsWith("Easy") = True Then
        Me.ReviewEasy1.Show()
        Me.ReviewEasy2.Show()
        Me.ReviewEasy3.Show()
        Me.ReviewMedium1.Hide()
        Me.ReviewMedium2.Hide()
        Me.ReviewMedium3.Hide()
        Me.ReviewHard1.Hide()
        Me.ReviewHard2.Hide()
        Me.ReviewHard3.Hide()
        Me.ReviewEasy1.Location = New Point((Me.Size.Width + 150) - Me.Size.Width, _
        (Me.Size.Height + 125) - Me.Size.Height)
        Me.ReviewEasy2.Location = New Point(Me.Size.Width - 200, (125 + Me.Size.Height) _
        - Me.Size.Height)
        Me.ReviewEasy3.Location = New Point((Me.Size.Width / 2), (Me.Size.Height - 125))
    ElseIf SessionBox.Text.EndsWith("Medium") = True Then
        Me.ReviewEasy1.Hide()
        Me.ReviewEasy2.Hide()
        Me.ReviewEasy3.Hide()
        Me.ReviewMedium1.Show()
        Me.ReviewMedium2.Show()
```

```
            Me.ReviewMedium3.Show()
            Me.ReviewHard1.Hide()
            Me.ReviewHard2.Hide()
            Me.ReviewHard3.Hide()
            Me.ReviewMedium1.Location = New Point(200, (Me.Size.Height / 2))
            Me.ReviewMedium2.Location = New Point(Me.Size.Width - 250, 150)
            Me.ReviewMedium3.Location = New Point((Me.Size.Width - 250), _
            (Me.Size.Height - 150))
        ElseIf SessionBox.Text.EndsWith("Hard") = True Then
            Me.ReviewEasy1.Hide()
            Me.ReviewEasy2.Hide()
            Me.ReviewEasy3.Hide()
            Me.ReviewMedium1.Hide()
            Me.ReviewMedium2.Hide()
            Me.ReviewMedium3.Hide()
            Me.ReviewHard1.Show()
            Me.ReviewHard2.Show()
            Me.ReviewHard3.Show()
            Me.ReviewHard1.Location = New Point(100, Me.Size.Height - 100)
            Me.ReviewHard2.Location = New Point(Me.Size.Width / 2, 100)
            Me.ReviewHard3.Location = New Point(Me.Size.Width - 100, Me.Size.Height - 100)
        End If
    End Sub

    'On window resize automatically move the image.
    Private Sub Review_Resize(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles Me.Resize
        Me.ReviewEasy1.Location = New Point((Me.Size.Width + 150) - Me.Size.Width, _
        (Me.Size.Height + 125) - Me.Size.Height)
        Me.ReviewEasy2.Location = New Point(Me.Size.Width - 200, (125 + Me.Size.Height) _
        - Me.Size.Height)
        Me.ReviewEasy3.Location = New Point((Me.Size.Width / 2 - 25), _
        (Me.Size.Height - 125))
        Me.ReviewMedium1.Location = New Point(200, (Me.Size.Height / 2))
        Me.ReviewMedium2.Location = New Point(Me.Size.Width - 250, 150)
        Me.ReviewMedium3.Location = New Point((Me.Size.Width - 250), (Me.Size.Height - 150))
        Me.ReviewHard1.Location = New Point(100, Me.Size.Height - 100)
        Me.ReviewHard2.Location = New Point(Me.Size.Width / 2, 100)
        Me.ReviewHard3.Location = New Point(Me.Size.Width - 100, Me.Size.Height - 100)
    End Sub

End Class
```

About.vb

```
Public Class About

    Private Sub About_Paint(ByVal sender As Object, ByVal e As _
    System.Windows.Forms.PaintEventArgs) Handles Me.Paint

        'Write the following text in the About window
        Dim heading As New Font("Arial", 16, FontStyle.Bold Or FontStyle.Underline)
        Dim subheading As New Font("arial", 12, FontStyle.Bold)
        Dim gtext As New Font("arial", 12, FontStyle.Regular)
        e.Graphics.DrawString("Rehabilitation Technologies", heading, Brushes.Black, 40, 10)
        e.Graphics.DrawString("Designer: ", subheading, Brushes.Black, 10, 50)
        e.Graphics.DrawString("Nathan Prasser", gtext, Brushes.Black, 90, 50)
        e.Graphics.DrawString("Version: ", subheading, Brushes.Black, 10, 80)
        e.Graphics.DrawString("1.0.0", gtext, Brushes.Black, 90, 80)
    End Sub
End Class
```

Help.vb

```
Public Class Help

    Private Sub Help_Paint(ByVal sender As Object, ByVal e As _
    System.Windows.Forms.PaintEventArgs) Handles Me.Paint

        'Write the following text in the help window
        Dim heading As New Font("Arial", 18, FontStyle.Bold Or FontStyle.Underline)
        Dim subheading As New Font("arial", 12, FontStyle.Bold)
        Dim gtext As New Font("arial", 12, FontStyle.Regular)
        e.Graphics.DrawString("Rehabilitation Technologies Help Center", heading, _
        Brushes.Black, 100, 10)
        e.Graphics.DrawString("This section will help you understand the operation of the _
       program, this program", gtext, Brushes.Black, 10, 50)
        e.Graphics.DrawString("has been designed around the use of the rehabilitation _
       equipment", gtext, Brushes.Black, 60, 65)
        e.Graphics.DrawString("DIFFICULTY: ", subheading, Brushes.Black, 10, 100)
        e.Graphics.DrawString("This difficulty option allows a range of different item _
       sizes to be used", gtext, Brushes.Black, 130, 100)
        e.Graphics.DrawString("POINTS: ", subheading, Brushes.Black, 10, 120)
        e.Graphics.DrawString("There are two options under this tab, Test Locations and _
       Random Locations", gtext, Brushes.Black, 130, 120)
        e.Graphics.DrawString("Test Locations will only move the image too one of three _
       locations, but it", gtext, Brushes.Black, 130, 135)
        e.Graphics.DrawString("will log all user movements for later referal. Random _
       locations will display", gtext, Brushes.Black, 130, 150)
        e.Graphics.DrawString("the image at any point but will not log any movments.", _
       gtext, Brushes.Black, 130, 165)
        e.Graphics.DrawString("SENSITIVITY: ", subheading, Brushes.Black, 10, 185)
        e.Graphics.DrawString("Allows the device sensitivity to be adjusted to suit the _
       users requirements", gtext, Brushes.Black, 130, 185)
        e.Graphics.DrawString("GRID: ", subheading, Brushes.Black, 10, 205)
        e.Graphics.DrawString("Allows a grid to be toggled on/off to assist in directional _
       interpretation", gtext, Brushes.Black, 130, 205)
        e.Graphics.DrawString("REVIEW: ", subheading, Brushes.Black, 10, 225)
        e.Graphics.DrawString("In this section the user can review all movements made _
       during Test Locations.", gtext, Brushes.Black, 130, 225)
        e.Graphics.DrawString("To do this, select a date from the drop down box and a _
       session from the other", gtext, Brushes.Black, 130, 240)
        e.Graphics.DrawString("drop down box, then select the go button. This selected _
       information will be", gtext, Brushes.Black, 130, 255)
        e.Graphics.DrawString("Displayed on the screen. Click the return button when _
       finished reviewing and", gtext, Brushes.Black, 130, 270)
        e.Graphics.DrawString("you will be returned to the original application window", _
       gtext, Brushes.Black, 130, 285)
        e.Graphics.DrawString("SHORT CUTS: ", subheading, Brushes.Black, 10, 305)
        e.Graphics.DrawString("Every function has the ability for shortcuts to be used. _
       Simply press the 'alt'", gtext, Brushes.Black, 130, 305)
        e.Graphics.DrawString("and you will notice a line underneath the first character _
       of each word in the", gtext, Brushes.Black, 130, 320)
        e.Graphics.DrawString("toolbar. Press the buttons associated with the commands you _
       wish to perform", gtext, Brushes.Black, 130, 335)
        e.Graphics.DrawString("Common short cut buttons are:", gtext, _
        Brushes.Black, 130, 350)
         e.Graphics.DrawString("Quit ", subheading, Brushes.Black, 130, 375)
         e.Graphics.DrawString("- alt+q", gtext, Brushes.Black, 216, 375)
         e.Graphics.DrawString("Review ", subheading, Brushes.Black, 130, 400)
         e.Graphics.DrawString("- alt+r", gtext, Brushes.Black, 216, 400)
         e.Graphics.DrawString("Difficulty ", subheading, Brushes.Black, 130, 425)
         e.Graphics.DrawString("- alt+d, e-m-h", gtext, Brushes.Black, 216, 425)
         e.Graphics.DrawString("Sensitivity ", subheading, Brushes.Black, 130, 450)
         e.Graphics.DrawString("- alt+s, l-m-h", gtext, Brushes.Black, 216, 450)
    End Sub
End Class
```

Section 2: Development Board Code

This following section lists the code written for the operation of the development board. Majority of this code was prewritten subroutines which were modified to handle this technology. This was done with the assistance of Scott Carden.

adc.c

```
#include "adc.h"
#include "delay.h"
#include "iolpc2138.h"

// Available Pins on Ext output are
/*
    External Header
  Ground          20        19  3.3V
  Out             18        17  P1_16 FREE
  P1_25 FREE   16        15  P1_24 FREE
  P0_29 AD0.2  14        13  P0_28 AD0.1
  P0_27 AD0.0  12        11  P0_26 AD0.5
  P0_19 FREE   10        9   P0_18 FREE
  P0_17 FREE    8        7   P0_13 AD1.4
  P0_12 AD1.3   6        5   P0_8 AD1.1
  P0_7 FREE       4        3   P0_6 AD1.0
  P0_5 AD0.7    2        1   P0_4 AD0.6
*/

void adcInit()
{
//Set 5 pins to input for ADC
  PINSEL0 = 0x00000F00; //Selects ADC's 0.6, 0.7
  PINSEL1 = 0x00100000; //Selects ADC 0.5
}

//------------------------- getAnalogueInput_AD0 function ---------------//
unsigned short int getAnalogueInput_AD0(unsigned char channel)
{
  unsigned short int val;

//AD0CR = 0x00200600; // enable ADC, 11 clocks/10 bits, ADC clock = 4.286 MHz

//start conversion (for selected channel)
// Delay(1000000); // Output fewer values
```

```
  AD0CR = 0x00200600 | (1 << channel);
  AD0CR |= 0x01000000;

  //wait until done
  while((AD0DR & 0x80000000) == 0);

  // get ADC data
  val = ((AD0DR & 0x0000FFC0)) >> 6;

  return(val);
}
```

adc.h

```
//adc.h
void adcInit(void);
unsigned short int getAnalogueInput_AD0(unsigned char channel);
```

buttons.c

```
//buttons.c
#include <iolpc2138.h>
#include "buttons.h"

// init buttons port
void InitButtons(void) {
  IO0DIR_bit.P0_15 = 0;   //set port0.15 as input (button 1)
  IO0DIR_bit.P0_16 = 0;   //set port0.16 as input (button 2)
  IO0DIR_bit.P0_20 = 0;   //set port0.20 as input (button 3)
  IO0DIR_bit.P0_30 = 0;   //set port0.30 as input (button 4)
  IO0DIR_bit.P0_9 = 0;    //set port0.9 as input (button 5)
}
```

buttons.h

```
// button.h
#include <iolpc2138.h>
#define BUT1_PRESSED   IO0PIN_bit.P0_15
#define BUT2_PRESSED   IO0PIN_bit.P0_16
```

```
#define BUT3_PRESSED   IO0PIN_bit.P0_20
#define BUT4_PRESSED   IO0PIN_bit.P0_30
#define BUT5_PRESSED   IO0PIN_bit.P0_9

// init buttons port
void InitButtons(void);
```

delay.c

```
//A simple delay
void Delay (unsigned long a) {
  while (--a!=0);
}
```

delay.h

```
void Delay(unsigned long a);
```

lcd.c

```
//lcd.c
#include <string.h>
#include "delay.h"
#include "lcd.h"

unsigned long data;

void E_Pulse()
{
  IO1SET_bit.P0_18 = 1; //set E to high
  //Delay(10);
  Delay(100);
  IO1CLR_bit.P0_18 = 1; //set E to low
}

void LCDInit()
{
  //first set D4, D5, D6, D7, RS, RW, E to output ports
  IO1DIR_bit.P0_17 = 1;
  IO1DIR_bit.P0_18 = 1;
  IO1DIR_bit.P0_19 = 1;
  IO1DIR_bit.P0_20 = 1;
  IO1DIR_bit.P0_21 = 1;
  IO1DIR_bit.P0_22 = 1;
  IO1DIR_bit.P0_23 = 1;

  //LCD initialization
  //step by step (from Gosho) - from DATASHEET

  IO1CLR_bit.P0_17 = 1;  //set RS port to 0
  IO1CLR_bit.P0_19 = 1;  //set R/W port to 0
  IO1CLR_bit.P0_18 = 1;  //set E port to 0

  Delay(110000);       //delay ~110ms
  //Delay(1100000);       //delay ~110ms

  IO1SET_bit.P0_20 = 1;   //set D4 port to 1
  IO1SET_bit.P0_21 = 1;   //set D5 port to 1
  E_Pulse();           //high->low to E port (pulse)
  Delay(10000);        //delay ~10ms
  //Delay(100000);        //delay ~10ms

  IO1SET_bit.P0_20 = 1;   //set D4 port to 1
  IO1SET_bit.P0_21 = 1;   //set D5 port to 1
  E_Pulse();           //high->low to E port (pulse)
  Delay(10000);        //delay ~10ms
  //Delay(100000);        //delay ~10ms

  IO1SET_bit.P0_20 = 1;   //set D4 port to 1
  IO1SET_bit.P0_21 = 1;   //set D5 port to 1
  E_Pulse();           //high->low to E port (pulse)
  Delay(10000);        //delay ~10ms
  //Delay(100000);        //delay ~10ms

  IO1SET_bit.P0_21 = 1;   //set D5 port to 1
  IO1CLR_bit.P0_20 = 1;   //set D4 port to 0
  E_Pulse();           //high->low to E port (pulse)
}
void LCDSendCommand(unsigned long a)
{
  IO1CLR_bit.P0_19 = 1;                  //set RW port to 0
```

```
 //Delay(2000);                  //delay for LCD char ~2ms
 Delay(20000);                   //delay for LCD char ~2ms
 data = 0x0;

 data = 0xffffff0f | a;          //get high 4 bits
 IO1CLR |= 0x00f00000;           //clear D4-D7
 data = data << 16;

 IO1SET = (IO1SET | 0x00f00000) & data;   //set D4-D7
 IO1CLR_bit.P0_17 = 1;           //set RS port to 0 -> display set to comand mode
 E_Pulse();                      //pulse to set d4-d7 bits

 //data = (a << 4) & 0x000000f0;          //get low 4 bits
 //IOCLR |= 0x000000f0;          //clear D4-D7
 //data = data <<4;
 data = 0x0;
 a = a<<4;

 data = 0xffffff0f | a;          //get high 4 bits
 IO1CLR |= 0x00f00000;           //clear D4-D7
 data = data << 16;
 IO1SET = (IO1SET | 0x00f00000) & data;   //set D4-D7
 IO1CLR_bit.P0_17 = 1;           //set RS port to 0 -> display set to command mode
 E_Pulse();                      //pulse to set d4-d7 bits
}
void LCDSendChar(unsigned long a)
{
 IO1CLR_bit.P0_19 = 1;           //set RW port to 0

 Delay(2000);                    //delay for LCD char ~2ms
 //Delay(20000);                 //delay for LCD char ~2ms
 data = 0x0;

 data = 0xffffff0f | a;          //get high 4 bits
 IO1CLR |= 0x00f00000;           //clear D4-D7
 data = data << 16;

 IO1SET = (IO1SET | 0x00f00000) & data;   //set D4-D7
 IO1SET_bit.P0_17 = 1;           //set RS port to 0 -> display set to comand mode
 E_Pulse();                      //pulse to set d4-d7 bits
 //data = (a << 4) & 0x000000f0;              //get low 4 bits
 //IOCLR |= 0x000000f0;                 //clear D4-D7
 //data = data <<4;
 data = 0x0;
 a = a<<4;

 data = 0xffffff0f | a;              //get high 4 bits
 IO1CLR |= 0x00f00000;                //clear D4-D7
 data = data << 16;
 IO1SET = (IO1SET | 0x00f00000) & data;    //set D4-D7
 IO1SET_bit.P0_17 = 1;                 //set RS port to 0 -> display set to command mode
 E_Pulse();                          //pulse to set d4-d7 bits
}
void LCDSendTxt(char* a)
{
 LCDSendCommand(CLR_DISP);
 LCDSendCommand(DD_RAM_ADDR);
 for(int i=0; i<strlen(a); i++)
 {
  LCDSendChar(a[i]);
 }
}
void LCDSendInt(int a)
{
 LCDSendCommand(CLR_DISP);
 LCDSendCommand(DD_RAM_ADDR);
 int h = 0;
 int l = 0;

 l = a%10;
 h = a/10;

 LCDSendChar(h+48);
 LCDSendChar(l+48);
}
void SmartUp(void)
{
 for(int i=0; i<40; i++) LCDSendCommand(CUR_RIGHT);
}
void SmartDown(void)
{
 for(int i=0; i<40; i++) LCDSendCommand(CUR_LEFT);
```

```
}

void Light(short a)
{
 if(a == 1)
 {
  IO0SET_bit.P0_21 = 1;
  IO0DIR_bit.P0_21 = 1;
  IO0SET_bit.P0_25 = 1;
  IO0DIR_bit.P0_25 = 1;
 }
 if(a == 0)
 {
  IO0SET_bit.P0_21 = 0;
  IO0DIR_bit.P0_21 = 0;
  IO0SET_bit.P0_25 = 0;
  IO0DIR_bit.P0_25 = 0;
 }
}
```

lcd.h

```
//lcd.h
#include <iolpc2138.h>

//#define       DISP_ON             0x0000000C    //LCD on
//#define       DISP_OFF      0x00000008    //LCD off
//#define       CLR_DISP      0x00000001    //LCD clear
//#define       CUR_HOME      0x00000002    //LCD cursor home
//#define       ENTRY_INC     0x00000007    //LCD increment
//#define       ENTRY_DEC     0x00000005    //LCD decrement
//#define       DD_RAM_ADDR         0x00000080    //LCD 1 row
//#define       DD_RAM_ADDR2        0x000000C0    //LCD 2 row
//#define       SH_LCD_LEFT   0x00000010    //LCD shift left
//#define       SH_LCD_RIGHT  0x00000014    //LCD shift right
//#define       MV_LCD_LEFT   0x00000018    //LCD move left
//#define       MV_LCD_RIGHT0x0000001C      //LCD move right

#define     CLR_DISP      0x00000001
#define     DISP_ON             0x0000000C
```

```
#define          DISP_OFF         0x00000008
#define          CUR_HOME         0x00000002
#define          CUR_OFF          0x0000000C
#define       CUR_ON_UNDER    0x0000000E
#define       CUR_ON_BLINK    0x0000000F
#define       CUR_LEFT       0x00000010
#define       CUR_RIGHT      0x00000014
#define          CUR_UP           0x00000080
#define          CUR_DOWN         0x000000C0
#define       ENTER         0x000000C0
#define          DD_RAM_ADDR           0x00000080
#define          DD_RAM_ADDR2          0x000000C0

void E_Pulse();
void LCDInit();
void LCDSendCommand(unsigned long a);
void LCDSendChar(unsigned long a);
void LCDSendTxt(char* a);
void SmartUp(void);
void SmartDown(void);
void Light(short a);
void LCDSendInt(int a);
main.c
```

main.c

```
/*************************************************************************
                    Header Files
*************************************************************************/
#include "system.h"
#include "buttons.h"
#include "lcd.h"
#include "delay.h"
#include "adc.h"
#include "rs232.h"
#include "stdio.h"
#define pi 3.14159265
/*************************************************************************
                       MAIN
*************************************************************************/
```

```
int main(void){

//************INITIALISATION************

//Initialise Variables
   int vMenu = 0;
   int vUpdate = 0;
   int u=0;
   float midpot =0;
   float endpot =0;

//Frequency Initialisation
   FrecInit();

//Initialise the buttons
   InitButtons();

//Initialise the adc
   adcInit();

// UART initialization
   UART0Initialize(9600);

//Initialise the LCD
   LCDInit();

//Turn the LCD on
   LCDSendCommand(DISP_ON);

//Turn the LCD light on
   Light(1);

//Write splash screen to LCD - 40 chars per line/16 viewable
   LCDSendTxt("StrokeTechnology                        Nathan Prasser");
//Pause for ~5sec to let user read info
   Delay(1000000);

//Update vMenu and vUpdate to jump to the first screen
   vMenu=1;
   vUpdate=1;

//Endless loop to check for button input to select 'program'
```

```
    while(1){

//Different selection for each potentiometer used
        /*      B1              Up
              B2  B3  B4      Left  Enter Right
                  B5            Down            */

//Which button was pressed and what does it do
//If Button 1 pressed
    if(!BUT1_PRESSED){
      vMenu--;
      vUpdate++;
//Control scrolling back past menu 1
        if (vMenu < 1){
          vMenu = 5;
        }
//Need ~0.1sec delay as system too fast
        Delay(100000);
    }

//************CALIBRATION BUTTON DETECTION************

//Update screen if a button was pressed
     if(vUpdate!=0){   //Check if need to change screen
//Update LCD to selection and run function
        switch(vMenu){
          case 1:
            LCDSendTxt("  HOLD BUTTON");
          default:
            break;
          }//End of switch
        //Reset vUpdate
        vUpdate = 0;
     }//End of if

//Get ADC results for sending by UART0
      char result[3];
      int j=6;
//run loop to check if button has been pushed
      if (u==0){
        int checkcount =0;
        int check = getAnalogueInput_AD0(5);
    int count;
//average values to remove ADC read error
    for (count = 1; count <= 5; count++){
     Delay(10000);
     checkcount = checkcount + getAnalogueInput_AD0(5);
    }

//if button has been pushed enter loop once only ever
//this is to declaire a startign point for calibration
    if (checkcount==0){
     u =1;
//calibrate midpot and end pot reading for calculations
     midpot = getAnalogueInput_AD0(6);
     endpot = getAnalogueInput_AD0(7);
//write b to rs232 to signify first button push
     UART0WriteChar('b');
     Delay(1000000);
//turn off display
     LCDSendCommand(DISP_OFF);
     Light(0);
    }
   }

//************MIDDLE POTENTIOMETER COMPUTATIONS************

//perminate loop once button pushed
   if (u==1){
// this is an attempt to remove the error within the POT
    int count =1;
    float num = 0;
//read midpot 100 times summing values
    for (count = 1; count <= 100; count++){
     Delay(100);
     num = num + getAnalogueInput_AD0(6);
    }
//find average value, done to reduce read error
    num = num/100;
//calibrated value subtracted from current value. allows movements
//left of calibrated point to be negative, and right to be positive
    num = midpot - num;

//Convert int result into char array to send over UART0
```

```
        sprintf( result, "%.2f", num );
        int k = 0;
        Delay(10000);
//send character "6" over rs232 to signify the following reading
//is from the middle pot (ADC0.6), for GUI data processing
        UART0WriteChar('6');
        Delay(10000);
//send ADC calculated value over rs232
        while (result[k]){
          UART0WriteChar(result[k]);
          k++;
        }
//allows end potentiometer loop to be entered
        u=2;
        Delay(100);
      }

//************END POTENTIOMETER COMPUTATIONS************

//exact same process as above, except ADC0.7 is read
      if (u==2){
        int count =1;
        float num = 0;
        for (count = 1; count <= 100; count++){
          Delay(100);
          num = num + getAnalogueInput_AD0(7);
        }
        num = num/100;
        num = endpot - num;
        sprintf( result, "%.2f", num );
        int i = 0;
        Delay(10000);
        UART0WriteChar('7');
        Delay(10000);
        while (result[i]){
          UART0WriteChar(result[i]);
          i++;
        }
        j++;
        u=1;
        Delay(100);
      }
```

```
//***********BUTTON DETECTION***********
   if (u==1 || u==2){
     int checkcount =0;
//Read ADC0.5 (button on handle) to detect button activation
     int check = getAnalogueInput_AD0(5);
     int count;
     for (count = 1; count <= 5; count++){
       Delay(10000);
       checkcount = checkcount + getAnalogueInput_AD0(5);
     }
     if (checkcount==0){UART0WriteChar('e');
       Delay(1000000);
     }
   }
 }//End of while
}//End of main

/***********************************************************************
          End of Program
***********************************************************************/
```

rs232.c

```
//rs232.c
#include "rs232.h"

unsigned int processorClockFrequency(void)
{
 //return real processor clock speed
 return OSCILLATOR_CLOCK_FREQUENCY * (PLLCON & 1 ? (PLLCFG & 0xF) + 1 : 1);
}

unsigned int peripheralClockFrequency(void)
{
 //VPBDIV - determines the relationship between the processor clock (cclk)
 //and the clock used by peripheral devices (pclk).
 unsigned int divider;
 switch (VPBDIV & 3)
 {
   case 0: divider = 4;  break;
   case 1: divider = 1;  break;
   case 2: divider = 2;  break;
 }
 return processorClockFrequency() / divider;
}

/**** UART0 ****/
void UART0Initialize(unsigned int baud)
{
 unsigned int divisor = peripheralClockFrequency() / (16 * baud);

 //set Line Control Register (8 bit, 1 stop bit, no parity, enable DLAB)
 U0LCR_bit.WLS   = 0x3;    //8 bit
 U0LCR_bit.SBS   = 0x0;    //1 stop bit
 U0LCR_bit.PE    = 0x0;    //no parity
 U0LCR_bit.DLAB  = 0x1;    //enable DLAB
 //with one row
 // U0LCR = 0x83;

 //devisor
 U0DLL = divisor & 0xFF;
 U0DLM = (divisor >> 8) & 0xFF;
 U0LCR &= ~0x80;

 //set functionalite to pins:  port0.0 -> TX0,  port0.1 -> RXD0
 PINSEL0_bit.P0_0 = 0x1;
 PINSEL0_bit.P0_1 = 0x1;
 //with one row
 //PINSEL0 = PINSEL0 & ~0xF | 0x5;
}
void UART0WriteChar(unsigned char ch0)
{
 //when U0LSR_bit.THRE is 0 - U0THR contains valid data.
 while (U0LSR_bit.THRE == 0);
 U0THR = ch0;
}
unsigned char UART0ReadChar(void)
{
 //when U0LSR_bit.DR is 1 - U0RBR contains valid data
 while (U0LSR_bit.DR == 0);
 return U0RBR;
```

```
}


unsigned char UART0ReadChar_nostop(void)
{
  //when U0LSR_bit.DR is 1 - U0RBR contains valid data
  if(U0LSR_bit.DR == 1) return U0RBR;
  else return 0;
}
void UART0WriteChar_nostop(unsigned char ch0)
{
  //when U0LSR_bit.THRE is 0 - U0THR contains valid data.
  if(U0LSR_bit.THRE == 1) U0THR = ch0;
}

/**** UART1 ****/
void UART1Initialize(unsigned int baud)
{
  unsigned int divisor = peripheralClockFrequency() / (16 * baud);

  //set Line Control Register (8 bit, 1 stop bit, no parity, enable DLAB)
  U1LCR_bit.WLS   = 0x3;    //8 bit
  U1LCR_bit.SBS   = 0x0;    //1 stop bit
  U1LCR_bit.PE    = 0x0;    //no parity
  U1LCR_bit.DLAB  = 0x1;    //enable DLAB
  //with one row
  // U0LCR = 0x83;

  //devisor
  U1DLL = divisor & 0xFF;
  U1DLM = (divisor >> 8) & 0xFF;
  U1LCR &= ~0x80;

  //set functionalite to pins:  port0.8 -> TX1,  port0.9 -> RXD1
  PINSEL0_bit.P0_8 = 0x1;
  PINSEL0_bit.P0_9 = 0x1;
  //with one row
  //PINSEL0 = PINSEL0 & ~0xF | 0x5;
}
void UART1WriteChar(unsigned char ch0)
{
  //when U0LSR_bit.THRE is 0 - U0THR contains valid data.
  while (U1LSR_bit.THRE == 0);
  U1THR = ch0;
}
unsigned char UART1ReadChar(void)
{
  //when U0LSR_bit.DR is 1 - U0RBR contains valid data
  while (U1LSR_bit.DR == 0);
  return U1RBR;
}
unsigned char UART1ReadChar_nostop(void)
{
  //when U0LSR_bit.DR is 1 - U0RBR contains valid data
  if(U1LSR_bit.DR == 1) return U1RBR;
  else return 0;
}
void UART1WriteChar_nostop(unsigned char ch0)
{
  //when U0LSR_bit.THRE is 0 - U0THR contains valid data.
  if(U1LSR_bit.THRE == 1) U1THR = ch0;
}
```

rs232.h

```
//rs232.h
#include <iolpc2138.h>

#define OSCILLATOR_CLOCK_FREQUENCY  14745600      //in MHz

//get real processor clock frequency
unsigned int processorClockFrequency(void);
//get peripheral clock frequency
unsigned int peripheralClockFrequency(void);

/**** UART0 ****/
//initialize UART0 interface
void UART0Initialize(unsigned int baud);
//write char to UART0 (RS232);
void UART0WriteChar(unsigned char ch0);
//read char from RS232
unsigned char UART0ReadChar(void);

//this function read/write char from RS232,
//but they not wait to read/write
unsigned char UART0ReadChar_nostop(void);
void UART0WriteChar_nostop(unsigned char ch0);


/**** UART1 ****/
//initialize UART0 interface
void UART1Initialize(unsigned int baud);
//write char to UART0 (RS232);
void UART1WriteChar(unsigned char ch0);
//read char from RS232
unsigned char UART0ReadChar(void);

//this function read/write char from RS232,
//but they not wait to read/write
unsigned char UART1ReadChar_nostop(void);
void UART1WriteChar_nostop(unsigned char ch0);
```

system.c

```
//system.c
#include <iolpc2138.h>
#include "system.h"
#define VIC_TIMER0_bit (1 << VIC_TIMER0)

//oscillator frequency
//IMPORTANT - if you use oscillator with different frequency,
//please change this value, becàuse timer not work correctly
#define OSCILLATOR_CLOCK_FREQUENCY  14745600      //in MHz

unsigned int GetCclk(void)
{
  //return real processor clock speed
  return OSCILLATOR_CLOCK_FREQUENCY * (PLLCON & 1 ? (PLLCFG & 0xF) + 1 : 1);
}
unsigned int GetPclk(void)
{
  //VPBDIV - determines the relationship between the processor clock (cclk)
  //and the clock used by peripheral devices (pclk).
  unsigned int divider;
```

```
 switch (VPBDIV & 3)
   {
    case 0: divider = 4;  break;
    case 1: divider = 1;  break;
    case 2: divider = 2;  break;
   }
 return GetCclk() / divider;
}
void FrecInit(void)
{
 //devide or multiplier
 //here is calculate frecuence
 PLLCFG_bit.MSEL = 0x2;  //M - multiplier
 PLLCFG_bit.PSEL = 0x1;  //P - devider
 //set changes (require from architecture)
 PLLFEED_bit.FEED = 0xAA;
 PLLFEED_bit.FEED = 0x55;

 //enable or connect PLL
 //enable PLL
 PLLCON_bit.PLLE = 1;
 //set changes (require from architecture)
 PLLFEED_bit.FEED = 0xAA;
 PLLFEED_bit.FEED = 0x55;
 //wait for PLOK (correct freq)
 while(PLLSTAT_bit.PLOCK == 0);
 //connect PLL
 PLLCON_bit.PLLC = 1;
 //set changes (require from architecture)
 PLLFEED_bit.FEED = 0xAA;
 PLLFEED_bit.FEED = 0x55;
}
```

system.h

```
//timer.h
// Initialize processor speed
void FrecInit(void);
// Get cclk
unsigned int GetCclk(void);
```

Appendix D: Potentiometer Technical Specifications

This appendix is information directly from the Farnell website. The pdf version of the data sheet for the 6187 potentiometer can viewed at the following link.
http://www.farnell.com/datasheets/103410.pdf

MODEL 6180 SERIES

7/8" Diameter
Single Turn
Conductive Plastic
Precision Potentiometer /
Position Sensor

MODEL STYLES

6181	1/8" Shaft, 1/4" Bushing
6186	1/8" Shaft, 3/8" Bushing
6187	1/4" Shaft, 3/8" Bushing

ELECTRICAL

Resistance Range, Ohms	1K to 100K
Standard Resistance Tolerance	±10%
Minimum Practical Resistance Tolerance	±5%
Independent Linearity *	±1.0%
Minimum Practical Independent Linearity	±0.5%
Input Voltage, Maximum	400Vdc not to exceed power rating
Power Rating, Watts	1.0 at 70°C derating to 0 at 125°C
Dielectric Strength	750V rms
Insulation Resistance, Minimum	1,000 Megohms
Output Smoothness, Maximum	0.1%
Actual Electrical Travel, Nominal	340° (300° with stop feature)
Electrical Continuity Travel, Nominal	350° (320° with stop feature)
End Voltage, Maximum	0.5% of input voltage
Resolution	Essentially Infinite
Temperature Coefficient**	-800 ppm/°C

ENVIRONMENTAL (MIL-R-39023)

Operating Temperature Range	Static: –65°C to +125°C Dynamic: –40°C to +125°C
Temperature Cycling	5 cycles, –65°C to +125°C (10% ΔR)
Shock, 6ms Sawtooth	100G's (0.1ms discontinuity max.)
Vibration	10G's, 10 to 500 Hz (2% ΔR, 0.1ms discontinuity max.)
Moisture Resistance	Five 24 hour cycles (25% ΔR)
High Temperature Exposure	1,000 hours at 125°C (0.5% ΔR)
Rotational Life	5 mil. shaft rev.
Rotational Load Life	5 mil. shaft rev. (10% ΔR)

MECHANICAL

Total Mechanical Travel	360° (320° ±3° with stop feature)
Number of Gangs, Maximum	1
Weight, Nominal (single gang)	0.53 oz.
Backlash, Maximum	1°
Static Stop Strength	40 oz.-in.
Panel Nut Tightening Torque, Maximum	25 lb.-in.
Start/Run Torque, Maximum	1.0 oz.-in.

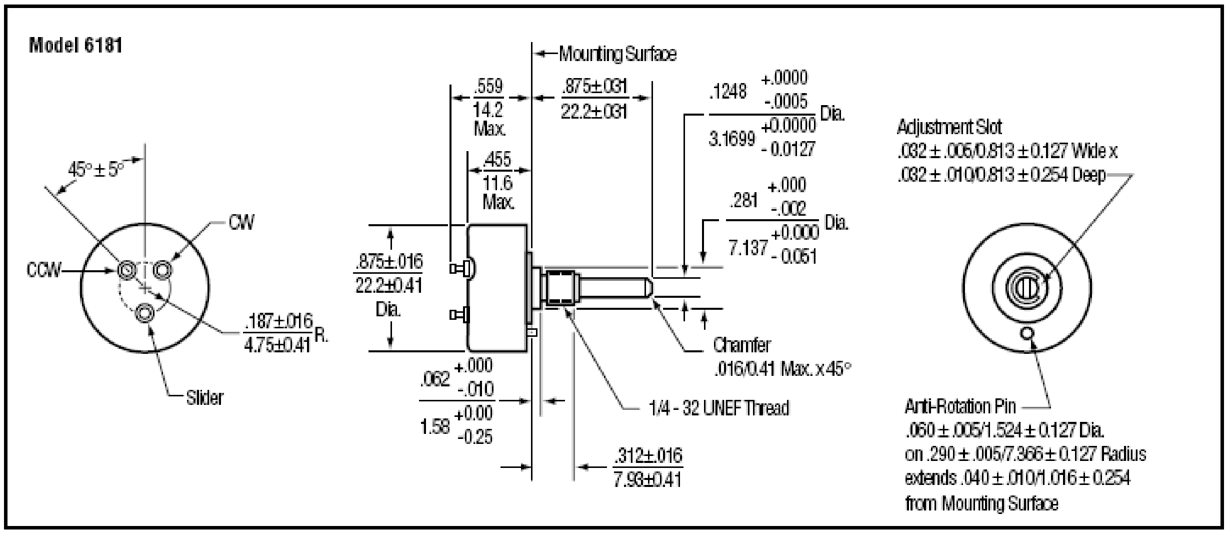
STANDARD RESISTANCE VALUES, OHMS

1K	2K	5K	10K	20K	50K

METRIC CONVERSIONS

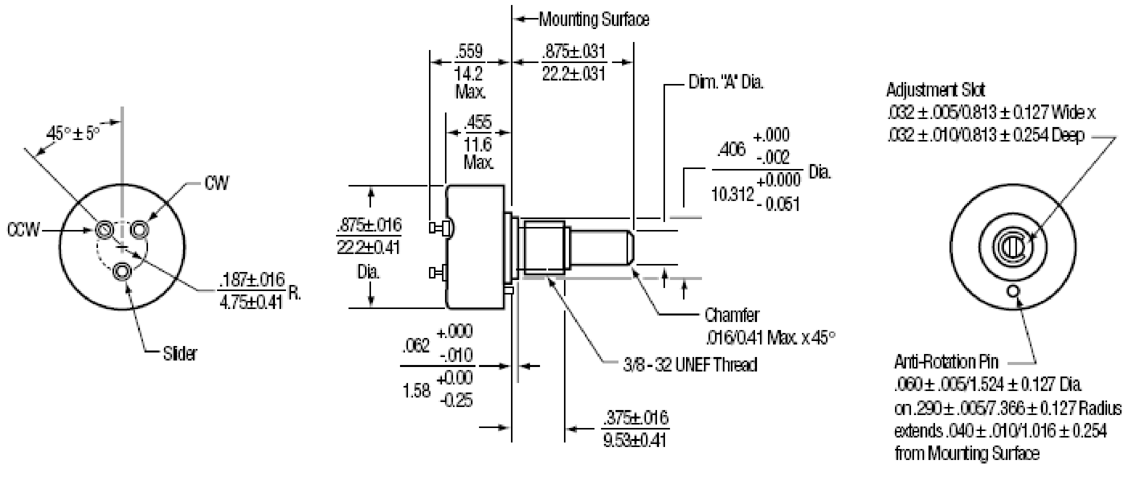
1 in.	25.4 mm	1 oz.-in.	0,007 N-m
1 oz	28.4 gm	1 lb.-in	0.113 N-m

OUTLINE DIMENSIONS (Inch/mm)


Model 6181
45° ± 5°
CW
CCW
.187±.016 / 4.75±0.41 R.
Slider
Mounting Surface
.559 / 14.2 Max.
.875±.031 / 22.2±0.31
.455 / 11.6 Max.
.875±.016 / 22.2±0.41 Dia.
.1248 +.0000 -.0005 / 3.1699 +0.0000 -0.0127 Dia.
.281 +.000 -.002 / 7.137 +0.000 -0.051 Dia.
Chamfer .016/0.41 Max. x 45°
.062 +.000 -.010
1.58 +0.00 -0.25
1/4 - 32 UNEF Thread
.312±.016 / 7.93±0.41
Adjustment Slot
.032 ± .005/0.813 ± 0.127 Wide x
.032 ± .010/0.813 ± 0.254 Deep
Anti-Rotation Pin
.060 ± .005/1.524 ± 0.127 Dia.
on .290 ± .005/7.366 ± 0.127 Radius
extends .040 ± .010/1.016 ± 0.254
from Mounting Surface

OUTLINE DIMENSIONS (Inch/mm)

Model 6186 & 6187



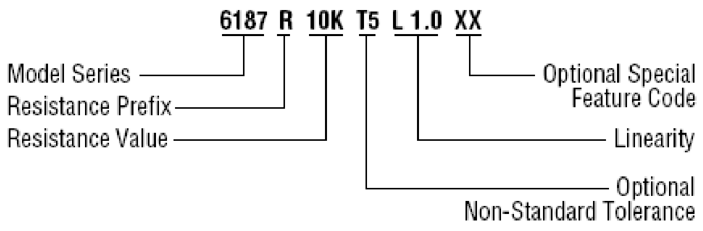
Mounting Surface
.559 14.2 Max.
.875±.031 22.2±.031
Dim. "A" Dia.
Adjustment Slot .032±.005/0.813±0.127 Wide x .032±.010/0.813±0.254 Deep
45°±5°
.455 11.6 Max.
.406 +.000 -.002 10.312 +0.000 -0.051 Dia.
CW
CCW
.875±.016 22.2±0.41 Dia.
.187±.016 4.75±0.41 R.
Slider
Chamfer .016/0.41 Max. x 45°
.062 +.000 -.010 1.58 +0.00 -0.25
3/8-32 UNEF Thread
.375±.016 9.53±0.41
Anti-Rotation Pin .060±.005/1.524±0.127 Dia. on .290±.005/7.366±0.127 Radius extends .040±.010/1.016±0.254 from Mounting Surface

Dim.	Model 6186	Model 6187
A	.1248 +.0000 -.0005 3.1699 +0.0000 -0.0127	.2497 +.0000 -.0005 6.3424 +0.0000 -0.0127

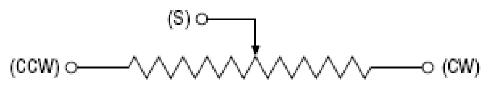
SPECIAL FEATURES

Center Tap	CT
Linearity Tape	LT
Shaft Lock	SL
Stop	ST

ORDERING INFORMATION



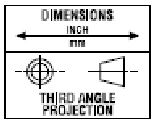
CIRCUIT DIAGRAM



NOTES

Metric equivalents, based on 1 inch = 25.4mm are rounded to the same number of significant figures as in the original English units and are provided for general information only.

Tolerances unless otherwise specified:
Linear = ± .01 inches (.25mm)
Angular = ± 2 degrees



Appendix E: LPC-MT-2138 Technical Details

This appendix is information directly from the guide for operation of the LPC-MT-2138. Complete information can be viewed at http://www.olimex.com/dev/pdf/MT-2138.chm

The **LPC2138** are based on a 32 bit ARM7TDMI-STM CPU with real-time emulation and embedded trace support, together with 512 kilobytes (kB) of embedded high speed flash memory. A 128-bit wide internal memory interface and a unique accelerator architecture enable 32-bit code execution at maximum clock rate. For critical code size applications, the alternative 16-bit Thumb Mode reduces code by more than 30% with minimal performance penalty.

Due to their tiny size and low power consumption, these microcontrollers are ideal for applications where miniaturization is a key requirement, such as access control and point-of-sale. With a wide range of serial communications interfaces and on-chip SRAM options of 16/32 kilobytes, they are very well suited for communication gateways and protocol converters, soft modems, voice recognition and low end imaging, providing both large buffer size and high processing power. Various 32 bit timers, single or dual 10-bit 8 channel ADC(s), 10 bit DAC, PWM channels and 47 GPIO lines with up to 9 edge or level sensitive external interrupt pins make these microcontrollers particularly suitable for industrial control and medical systems.

The **LPC2138-MT** Development board is designed to evaluate LPC2138 processor. It has the following features:

- JTAG connector as per ARM's 2x10 pin layout, ARM-JTAG (Wiggler) compatible
- 14.7456 MHz crystal allow easy communication setup
- RS232 interface circuit with SUB-D 9 pin connector
- LCD16x2 display
- five buttons connected to interrupts ports
- Dallas iButton port
- Frequency input
- Relay with 10A/250VAC contacts
- Buzzer
- Status LED
- RESET button
- Bootloader enable jumper and pullup
- DEBUG jumper for JTAG enable/disable
- Power plug-in jack
- two on board voltage regulators 3.3V and 5V
- power supply filtering capacitor
- Four mounting holes
- PCB: FR-4, 1.5 mm (0,062"), green soldermask, white silkscreen component print
- Dimensions: 120x38 mm (4.75x1.5")

OLIMEX LPC2138-MT Board

LPCP2138-MT Hardware description

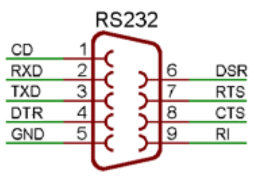
Peripherials

Unit	Description
LCD Display	2X16 LCD Display
COM Port2	RS232 DB9 Female connector for LPC2138 UART1.
JTAG Connector	2x10 0,1" step connector for programming with ARM-JTAG.
I2C Connector	4 pins connector for devices which use I2C protocol.
Frequency Connector	External frequency connector which is connected to P0.10 (pin 35)
Dallas Connector	Interface to Dallas device connected to P0.11 (pin 37).
Battery Connector	External battery connector
Buttons	Five buttons connected to interrupt ports: Button1 - P0.15 (pin 45), Button2 - P0.16 (pin 46), Button3 - P0.20 (pin 55), Button4 - P0.30 (pin 15), Button5 - P0.9 (pin 34)
Led	Red status led connected to P0.31 (pin 17)

Technical characteristics

Parameter	Description
Voltage Supply	12.0V DC 9.0V AC
CPU	LPC2138F
Crystals	Q1 - 14,745 MHz HF crystal Q2 - 32,768 KHz clock crystal
Board dimensions	120x38 mm (4.75x1.5")
PCB	FR-4, 1.5 mm (0,062"), green soldermask, white silkscreen component print
Operating Temperature	form 0ºC to 70ºC

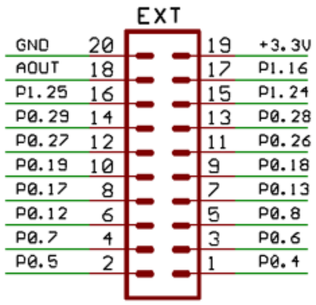
RS232 Connector



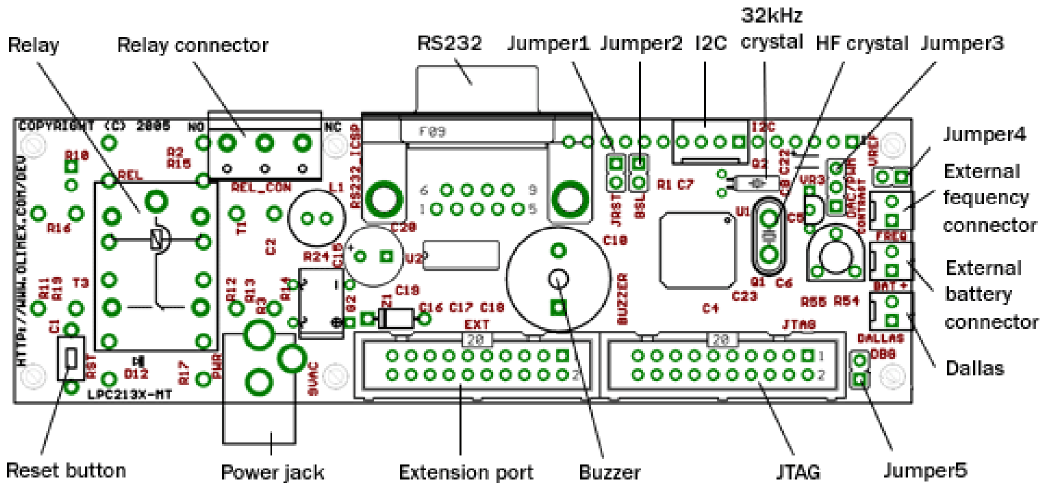
Pin / Name	Connected to:	Functionality
1 - CD	not connected	-
2 - TXD	PIN 19	P0.0 / TXD0 / PWM1
3 - RXD	PIN 21	P0.1 / RXD0 / PWM3 / EINT0
4 - DTR	MAX3232	-
5 - GND	GROUND	-
6 - DSR	not connected	-
7 - RTS	not connected	-
8 - CTS	not connected	-
9 - RI	not connected	-

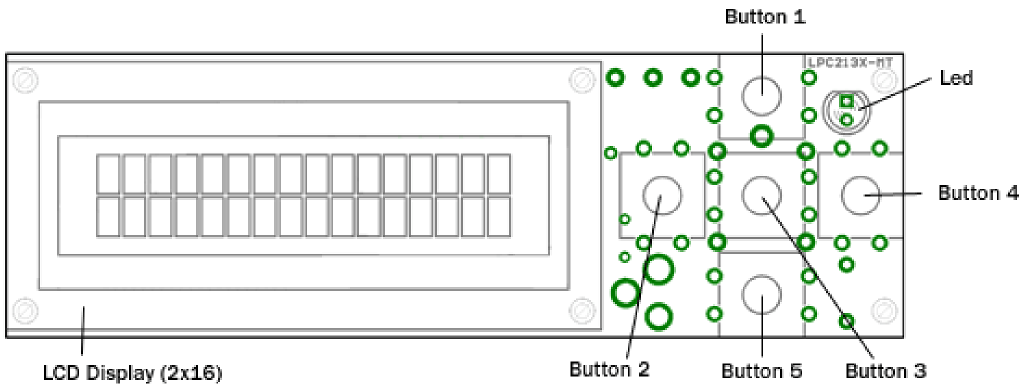
Max communication baud rate supported by LPC2138F is 448 kbps, maximum baud rate with MAX3232 driver on board is 250 kbps.

Extension port

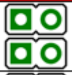
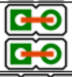
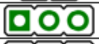
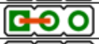
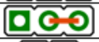
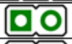
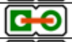
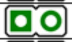
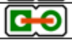


Pin / Name	Connected to:	Functionality
1 - P0.4	PIN27	P0.4 / SCK0 / CAP0.1 / AD0.6
2 - P0.5	PIN29	P0.5 / MISO0 / MAT0.1 / AD0.7
3 - P0.6	PIN30	P0.6 / MOSI0 / CAP0.2 / AD1.0
4 - P0.7	PIN31	P0.7 / SSEL0 / PWM2 / EINT2
5 - P0.8	PIN33	P0.8 / TXD1 / PWM4 / AD1.1
6 - P0.12	PIN38	P0.12 / DSR1 / MAT1.0 / AD1.3
7 - P0.13	PIN39	P0.13 / DTR1 / MAT1.1 / AD1.4
8 - P0.17	PIN47	P0.17 / CAP1.2 / SCK1 / MAT1.2
9 - P0.18	PIN53	P0.18 / CAP1.3 / MISO1 / MAT1.3
10 - P0.19	PIN54	P0.19 / MAT1.2 / MOSI1 / CAP1.2
11 - P0.26	PIN10	P0.26 / AD0.6
12 - P0.27	PIN11	P0.27 / AD0.0 / CAP0.1 / MAT0.1
13 - P0.28	PIN13	P0.28 / AD0.1 / CAP0.2 / MAT0.2
14 - P0.29	PIN14	P0.29 / AD0.2 / CAP0.3 / MAT0.3
15 - P1.24	PIN32	P1.24 / TRACECLK
16 - P1.25	PIN28	P1.25 / EXTIN0
17 - P1.16	PIN16	P1.16 / TRACEPKT0
18 - AOUT	PIN 9	P0.25 / AD0.4 / AOUT
19 - +3.3V	+3.3V	-
20 - GND	GROUND	-





Jumpers

Jumpers	Position	Description
Jumper 1 (JRST) Jumper 2 (BSL)		Disable ICSP programming
		Enable ICSP programming (via RS232 Connector)
Jumper 3 (DAC/PWM)		LCD light is not control.
		LCD light is control from PWM5 port (pin 1).
		LCD light is control from CAP and connected to AOUT (port 9)
Jumper 4 (VREF)		VREF port is not connected.
		VREF port is connected to 3.3V
Jumper 5 (DBG)		Disable JTAG programming.
		Enable JTAG programming.

Appendix F: Complete Set of Results

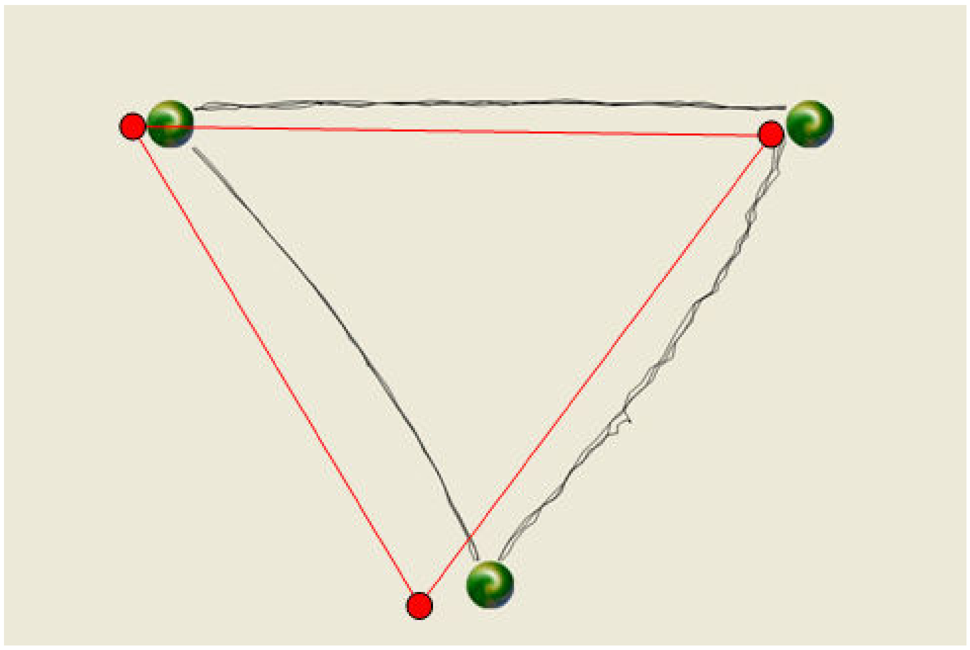


Figure Appendix F.1: Result 1 – Easy

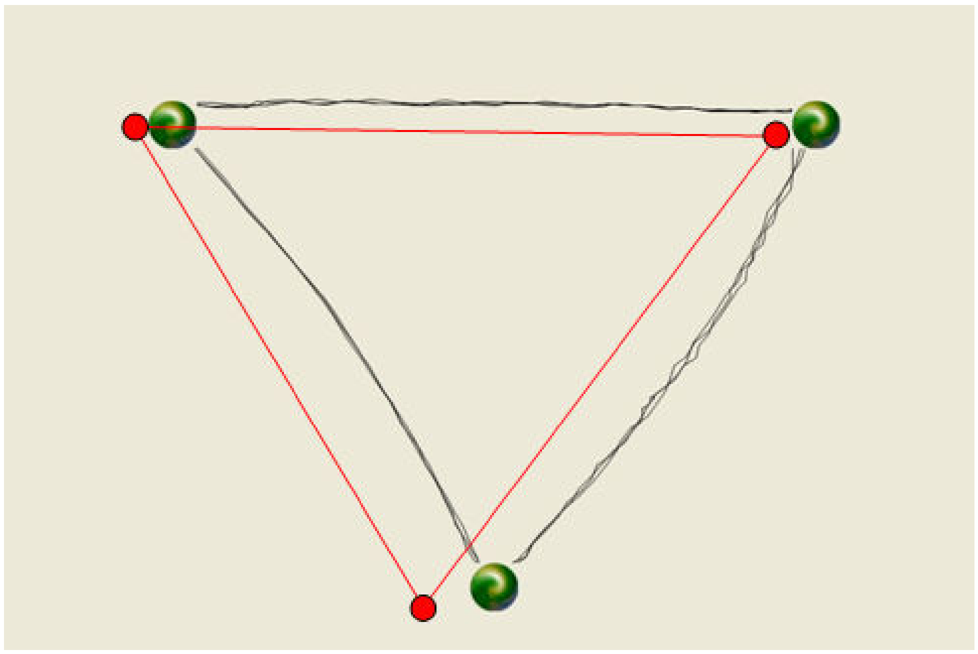


Figure Appendix F.2: Result 2 – Easy

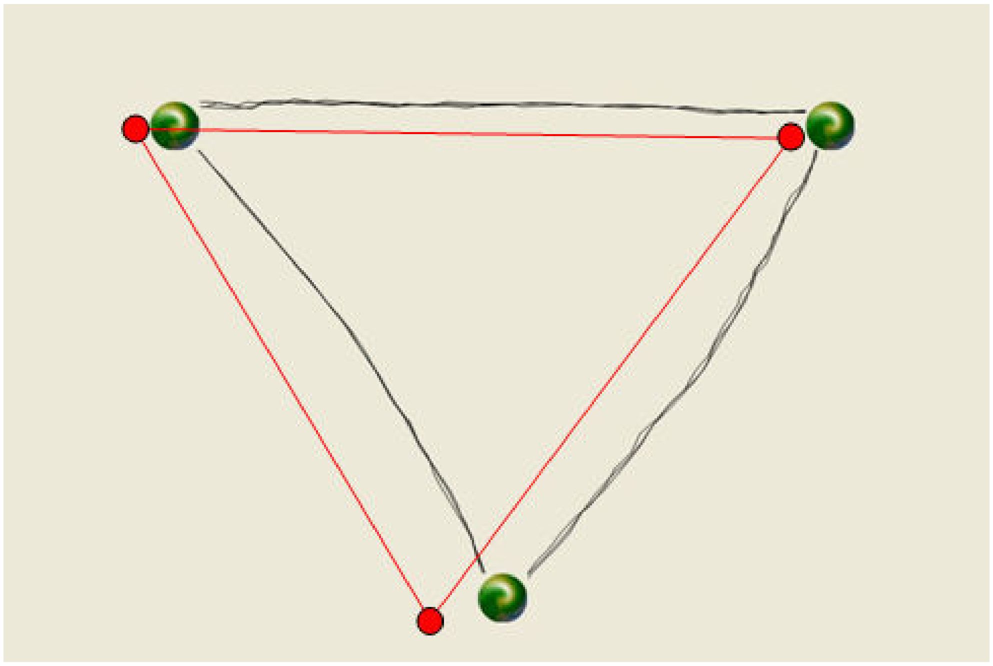


Figure Appendix F.3: Result 3 – Easy

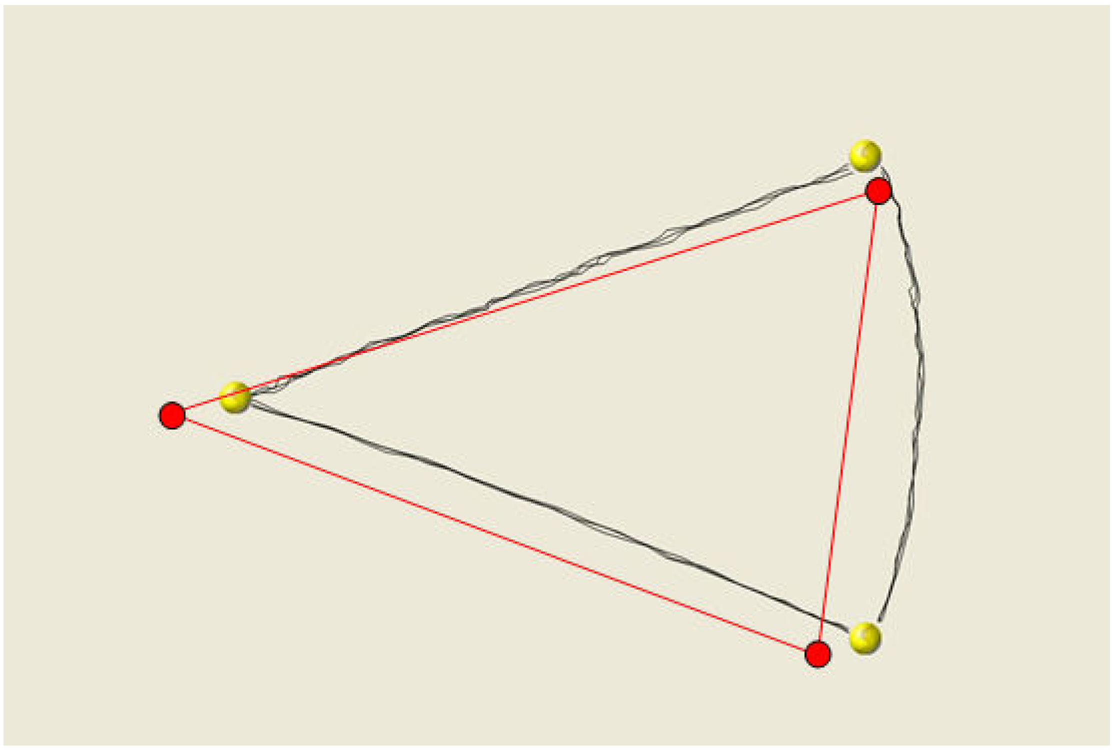


Figure Appendix F.4: Result 1 – Medium

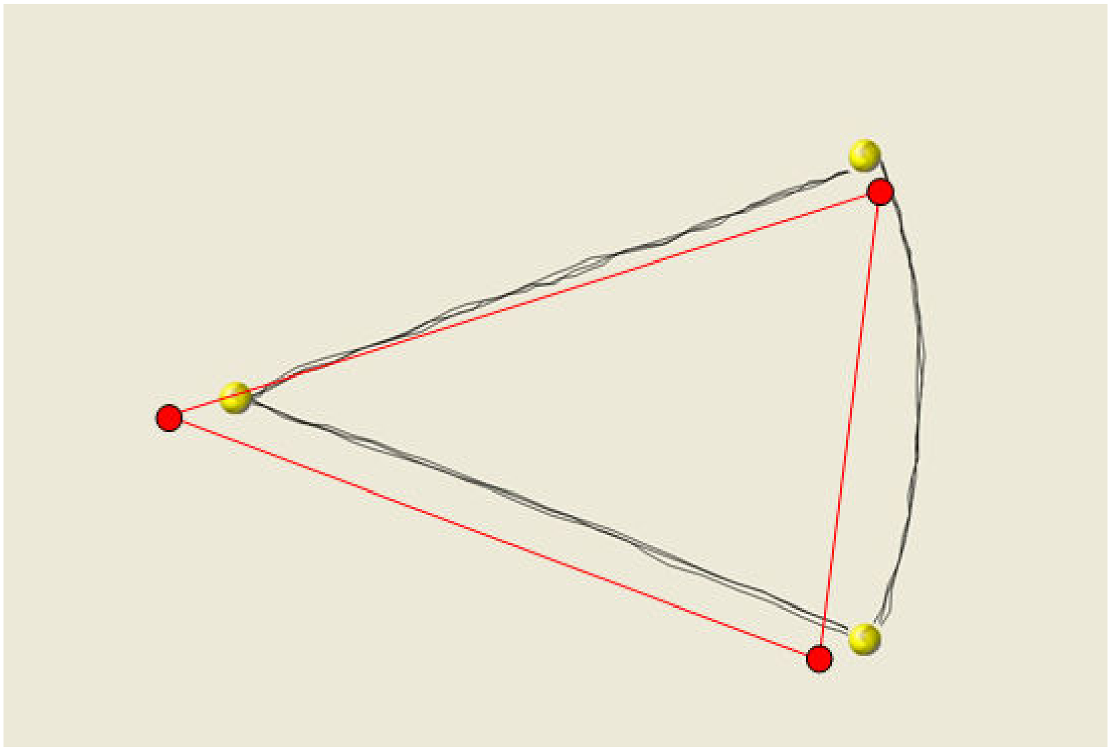


Figure Appendix F.5: Result 2 – Medium

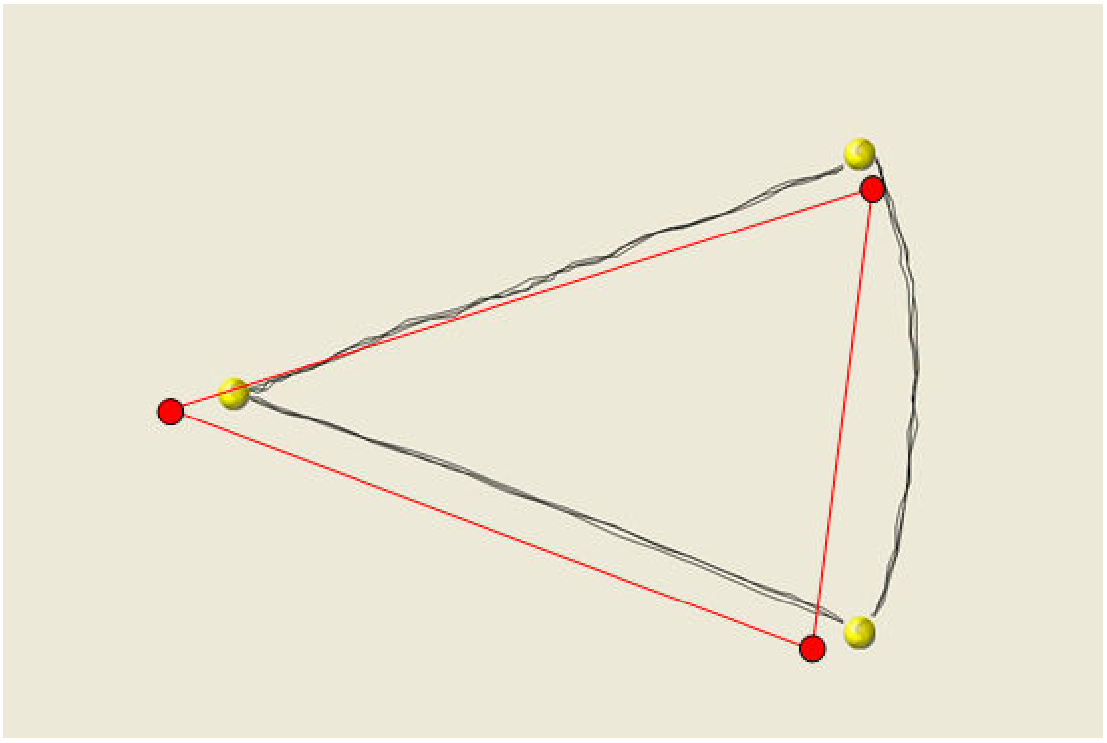


Figure Appendix F.6: Result 3 – Medium

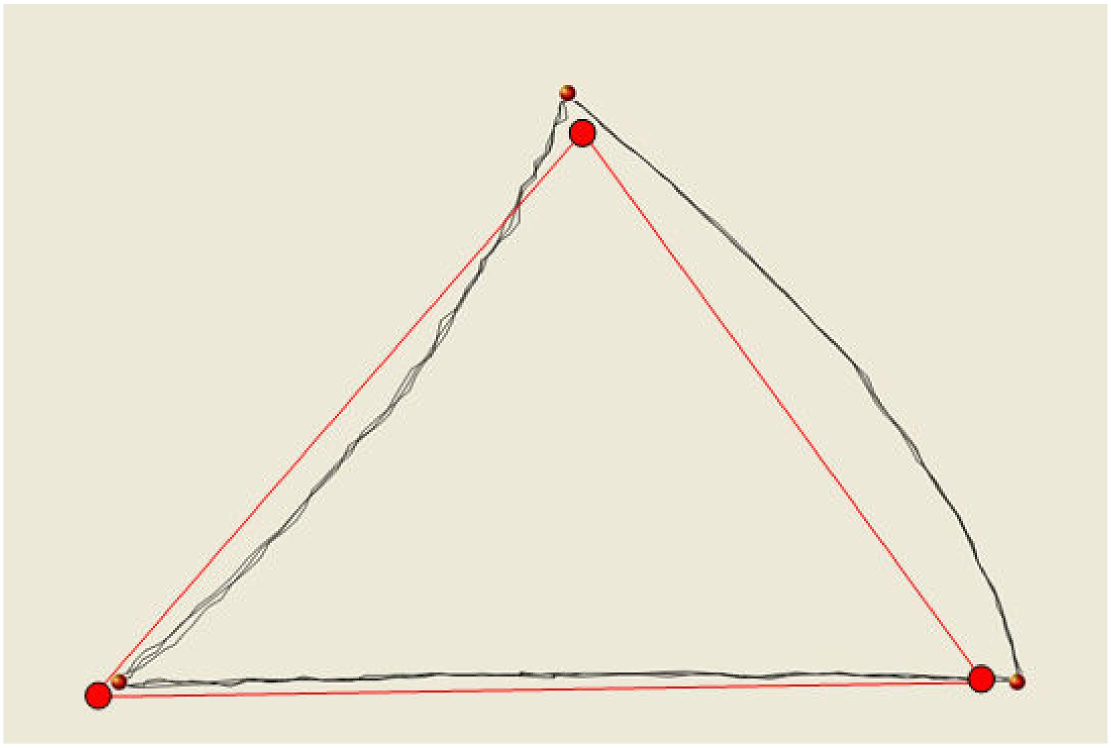


Figure Appendix F.7: Result 1 – Hard

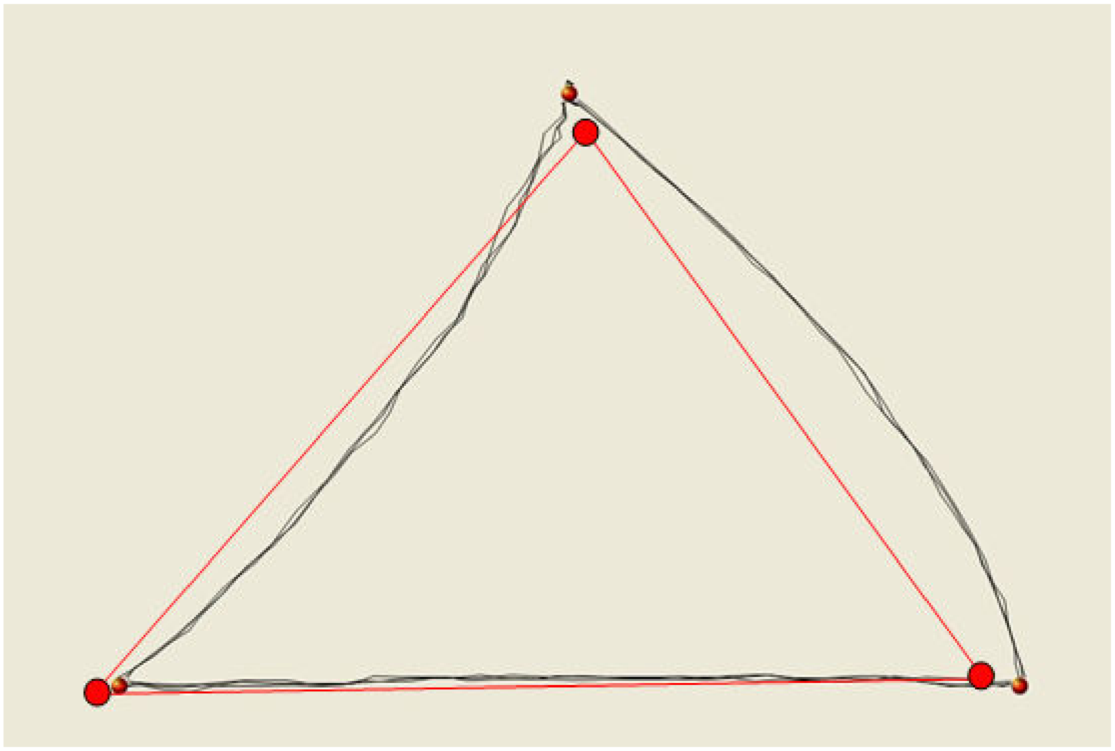


Figure Appendix F.8: Result 2 – Hard

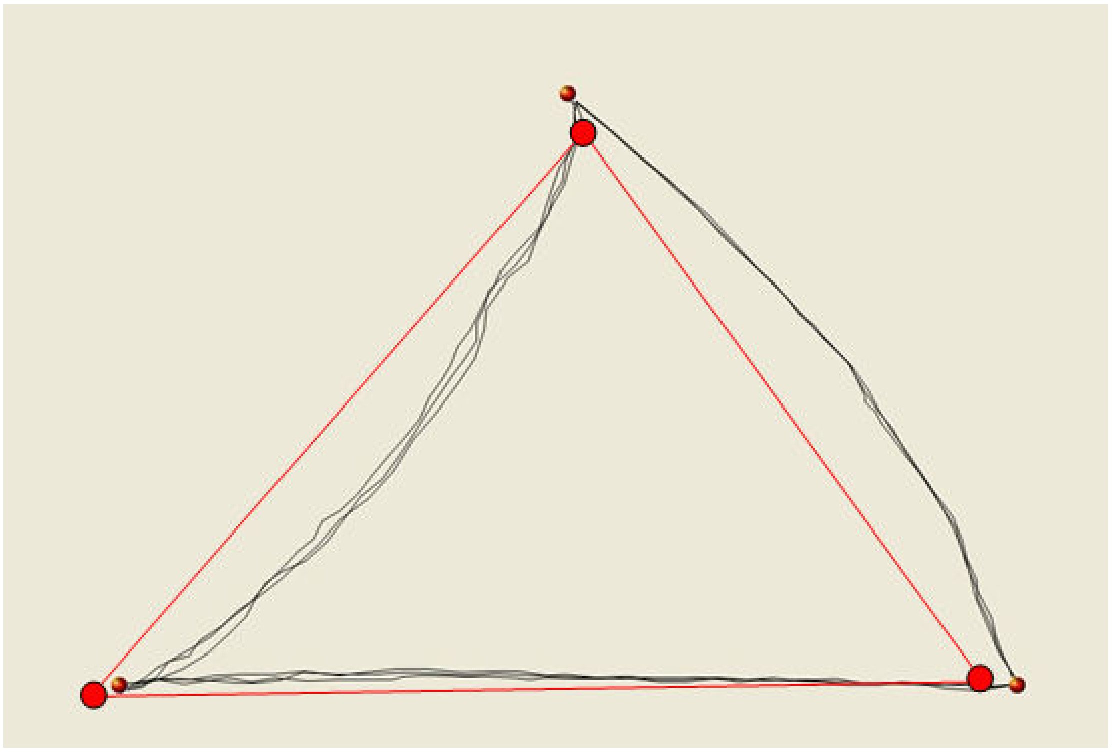


Figure Appendix F.9: Result 3 – Hard