A PLANAR CABLE-DRIVEN ROBOTIC DEVICE FOR PHYSICAL THERAPY ASSISTANCE

by

Melissa M. Morris

A Thesis Submitted to the Faculty of

The College of Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

Florida Atlantic University

Boca Raton, Florida

December 2007

A PLANAR CABLE-DRIVEN ROBOTIC DEVICE FOR PHYSICAL THERAPY ASSISTANCE

by
Melissa M. Morris

This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Oren Masory, Department of Mechanical Engineering, and has been approved by the members of her supervisory committee. It was submitted to the faculty of The College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:

Dr. Oren Masory
Thesis Advisor

Dr. Daniel Raviv

Dr. Zvi Roth

Dr. Oren Masory
Chairperson, Department of Mechanical Engineering

Dr. Karl Stevens
Dean, College of Engineering and Computer Science

Dr. Barry Rosson
Dean, Graduate Studies

11-16-07
Date

ACKNOWLEDGMENTS

There are certainly many people who have helped me in attaining this achievement. Without their help, this work would not have been as easy and may not have been successful. Though I will surely leave many names out, I would like to give thanks to a few people who stand out the most.

The idea and fundamentals of using cables was inspired by some initial work with Dr. Ming Huang. The advice and suggestions given by Dr. Torok, Dr. Whitehurst and Dr. Stout were the first steps for the physical realization of this project as a physical training device.

Mark Royer and Jeff Webb provided assistance by carrying out the more difficult machining work so that the project really came together. Hank Vansant was extremely helpful in locating many electrical components and hardware that was vital to completing this project. Tom Kelly provided much-needed suggestions early on and was the greatest mentor to me during the early part of my graduate work.

I had the unique chance to collaborate on a similar project outside of Florida Atlantic University. Dr. Moshe Shoham, his team and especially his graduate student, Yonatan Wischnitzer, at the Technion Israel Institute of Technology provided me with different ideas and an outlook that helped me during the later stages of my work.

Dr. Zvi Roth not only helped to facilitate my visit to Israel, but also offered valuable suggestions as a member of my thesis committee. Dr. Daniel Raviv completed my committee and aided me in several areas of this project.

Dr. Oren Masory served as my greatest mentor in this work. As my advisor, he was always willing to give me assistance and ideas whenever they were needed. He gave me the exceptional support I needed to get through the challenges that arose. Most of the ideas that made this project so unique and functional were provided by him. Without his guidance, this project wouldn't have come into fruition.

Finally, I would like to thank Danny for being my supportive husband through the long process of finishing this work. In addition to helping with some of the construction and soldering, he supported me in every way I needed. He was there from start to finish of this endeavor. He gave me the needed encouragement to keep on going when things didn't seem they would work out. Though he aided me as an engineer, he was most helpful as a friend.

ABSRACT

Author: Melissa M. Morris

Title: A Planar Cable-Driven Robotic Device for Physical Therapy Assistance

Institution: Florida Atlantic University

Thesis Advisor: Dr. Oren Masory

Degree: Master of Science

Year: 2007

The design and construction of a tri-cable, planar robotic device for use in neurophysical rehabilitation is presented. The criteria for this system are based primarily on marketability factors, rather than ideal models or mathematical outcomes. The device is designed to be low-cost and sufficiently safe for a somewhat disabled individual to use unsupervised at home, as well as in a therapist's office. The key features are the use of a barrier that inhibits the user from coming into contact with the cables as well as a "break-away" joystick that the user utilizes to perform the rehabilitation tasks. In addition, this device is portable, aesthetically acceptable and easy to operate. Other uses of this system include sports therapy, virtual reality and teleoperation of remote devices.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

Human interaction with robots is becoming increasingly important as robotics is being applied in more areas of our lives. From traditional applications such as painting vehicles, robots have slowly moved into our homes as toys and appliances such as vacuum cleaners. As technology and user-acceptance increases, home robotics can take on more challenging tasks. There is currently only minimal contact between user and robot, such as the issuing of commands. Robots are being used as tools for extending the capabilities of the human beings, but rarely under autonomous function. Some more recent robotic designs, however, involve more closer and lengthy contact between an individual and the machine. One such area where this is the case is in assistive robotics.

Physical therapy and other types of muscle training require motions to be performed repetitively. Robots excel at repetitive tasks which makes them a perfect tool in these endeavors. By creating a platform by which a human and robot can interact, the robot can direct and evaluate a patient's response (motions and forces) in physical therapy with little involvement of a trainer. This is the major goal of robots such as the one described in this thesis.

1.1 Objectives

The goal of this project is to design a robot capable of being used as a tool in physical therapy, particularly of the upper limbs in planar motions. In order to effectively complete this task, the system should be capable of the following tasks:

- Be able to accurately guide the user (patient) through the paths repetitively
- Be able to gently guide the user through a predefined path
- Be able to provide various degrees of assistance to the user as the user's competence increases, including the possibility of providing a counter-force if the user moves in the wrong direction
- Be able to measure the progress of the patient
- Be adaptive by allowing for a programmable load along the trajectory and/or as a function of progress

and above all

- To safely do all of the above tasks without harm to the user

This particular project also dictated a few other desired qualities that are not necessarily inherent in all haptic devices for this use. While devices currently exist on the market for hospital and institution use, there are not any that are promising for home use. To make this device more attractive for home use, additional desired characteristics are sought:

- For the device to be inexpensive (affordable for home use or rental)
- To be easy to manufacture and repair (to minimized down-time and operation costs)
- To be flexible (both in the path programming and in the possibilities for its use)
- For the device to be user friendly
- To be able to be monitored by a remote location such as a physician's office for feedback, error diagnosis and reprogramming
- To be aesthetic and minimal usage of space in a home environment
- To be durable against minor abuse typical in a home setting
- To have additional safety features for child and pet interference

Though not required, one goal to strive for is to make the device somewhat entertaining. This would not only aid in its acceptance and user-friendliness, but also make it more likely that a home rehabilitation routine will be followed frequently without the presence of a therapist in the home.

1.2 Overview

This thesis will describe the design, construction and analysis of a robotic device for use as a tool for physical therapy. To this end, first this thesis provides a brief overview of the background and current progress in the use of robotics for rehabilitation and physical training. The possibilities for a system meeting the criteria set for this project are then explored, with the final design decision described. This is all provided in the next chapter.

The overall design of the system is outlined in Chapter 3. This includes discussion of aspects of the system layout, the kinematics involved and the operational modes. It is here that some of the unique aspects of this robot are presented.

To complete the theoretical design, control algorithms are detailed for the operation of the machine. Chapter 4 also includes details on the device operation from a controls perspective.

Chapter 5 describes the actual implementation of the design. The fabrication and specifications for the hardware begin this chapter. The final design and realization of the electrical components are also detailed. Actual implementation of the control algorithms in the software is also explained. Finally, the achievement of a simple user-friendly human interface is illustrated.

The evaluation of the results is presented in Chapter 6. This includes a description of the performance and a discussion on the limitations and requirements for use of this device.

Finally, the last chapter provides a summary of the work and possibilities for the future.

Chapter 2

BACKGROUND

Robotic devices have slowly become more common in everyday life. They are no longer confined solely to perform repetitive labor in large manufacturing facilities. Currently, integration of robotics and biological systems is a fast-growing area of study. While some researchers are focusing on using biology to inspire and influence robotic design [2] or the use of robotics to understand how the mind functions [28, 43, 50], much research is focused on using robotics to aid biological systems. Artificial limbs and organs, as well as robotic surgery equipment are some examples of devices that have been developed. Robotic therapeutic aids are another example of the fruits of this research. This particular application will be the focus of the remainder of this thesis.

Physical therapy robots come in two main forms, those that are used for assistance for daily living and those that are used in physiotherapy. Assistive robots include advanced prosthetics [10, 32] and aids for mobility and daily tasks [21, 34]. For a long time, these were the only applications for robotics in rehabilitation [41]. Over the past decade or so, the use of robotics in neurorehabilitation has become a reality. In this application, robots assist in the physiotherapy of individuals who suffer from disabilities due to stroke, Parkinson's Disease, Multiple Sclerosis, spinal cord injuries or other motor-impairment conditions. Where traditional therapy may last only a short time due to the high cost and limited availability of physicians, patients can experience further improvements if the therapy were to be continued [43]. With the amount of time permitted for rehabilitation by United States insurance companies decreasing, it is necessary to find ways to make the rehabilitation time more effective [7] or to allow patients to continue on their own and inexpensively. The use of robotics makes long-term rehabilitation a possibility – both by allowing patients to work without the constant aid of a therapist and by reducing the costs by performing the routines at home. The savings associated in the reduced number of trips to a therapist office, both monetary and in time, could further help to offset the initial cost of a home device.

The device cost, with proper design, could be made to be reasonable. Unlike assistive robots which are often tailored to a specific individual and problem, physical therapy devices would be ideal for many people. Mid-sized production possibilities of these devices may make the commercial implementation easier than that of the limited-production assistive devices [8].

Another important factor in the design and marketability of robotic therapy devices is the user acceptance of such devices. Early in the research of physical therapy robotics, it was found that there was some apprehension toward using robots, from both patients and their therapists [12]. After using the robotic aides, however, the apprehension mostly dissipated. Understanding of robotics was an important factor in acceptance of their use. Other research has shown that the use of robotics is readily accepted by patients, but not necessarily by therapists. While patients may work harder and enjoy simple exercises more, therapists may find that they themselves are bored or have less control. While this may be the case, the therapists can gain the ability to handle multiple patients at one time [16]. This aspect is less of a focus in current research tends. With the availability of robots that perform mundane chores such as vacuuming and mowing, as well as robots already in use at some hospitals aiding in surgery, the population in general is more accepting of robotic use. Still, some current research does point out that some users, particularly the elderly, feel that they can not cope with high-tech devices [8]. With some training with their therapist, as well as plenty of user-friendly features, these devices should be tolerable by most users, if not readily accepted. Robots are now gaining a reputation for being "fun" and also allow for more interesting exercises to be performed. Integrated games and challenges in particular should not only aid in the acceptance of these devices to older users, but make the interface very familiar to younger users that have grown up with toy robots and videogames. This flexibility and variety could make the device adjustable to interact with different people, whose unique personalities cause them interact with machines in diverse ways [26, 48]. The device could also have features to deflect aggravation and improve emotional response of the user [54].

The validity of the use of robotics in therapeutic improvement has been investigated by several research groups. The use of repetitive motions to regain motor capabilities has long been used and has been shown to be helpful, at least in the case of stroke. Many therapy techniques (traditional and robotic) use active-assistance where the patient tries to perform a task and is aided, as needed, to the completion. This type of therapy is the most predominate in research, possibly because it is the most natural for robotic implementation [43]. Another type of technique used is the active-constrained mode of learning. In this technique, a user's movement is halted if a deviation from a correct movement is made. Using any technique, early therapy is accepted as being better [9]. In general, it is agreed that long-term rehabilitation continues to improve the condition of the patient. However, after the first several months, the results are not as evident and are slower. Robot-aided neurorehabilitation in particular has been shown to indeed affect motor learning [28] and motor capacity [31] through studies using the MIT-MANUS robot (this device is detailed later). There is, however, some debate as to whether robotic-aided therapy is significantly better than traditional therapy [43]. It is clear that robotic-aided therapy works, but depending upon the system it may not work better than techniques that did not use robots as a tool. Robotic devices must provide the right type and the right amount of assistance to be as effective as or more effective than human therapists alone. This perfect treatment varies depending upon the individual patient as well the severity and nature of the impairment.

The issue of whether robotic-aided rehabilitation is effective when used in-home has not yet been determined. One study reviewed several studies on home-based rehabilitation and found that the outcome was similar to that of in-office care [5]. This studied focused on the costs, however and did not involve the use of robotics for any of the rehabilitation care. Still, if a robotic device can mimic the quality of care traditionally received during home-based sessions, there is no reason to believe that there would be any noticeable reduction in the progress made via robotic home-based rehabilitation.

One unique benefit to robotic physical therapy is the quantitative feedback on progress that can be determined. Traditional physical therapy tends to use somewhat objective measures in determining a patient's current level of performance and progress [46]. Even those that are quantitative (such as speed and strength of motion) may not be accurately and consistently measured by all therapists. While most rehabilitation robots can also perform measurement tasks, there are a few systems designed solely for the purpose of obtaining quantitative information of motor ability, such as that after spinal cord injuries [14].

2.1 Survey of Therapeutic Training Robotics

There are a variety of physical therapy robots currently being researched. Robots can be used in a variety of ways for a variety of rehabilitative needs [16]. One early system was designed simply to advance continuous passive motion machines to allow joints to be moved after surgery, allowing for better healing. This involved two simple robotic arms with planar motion capabilities working in unison to properly flex the joint [27], as shown in Figure 2-1. Robotic advances such as increased computing power now allow for more intricate systems to be created.

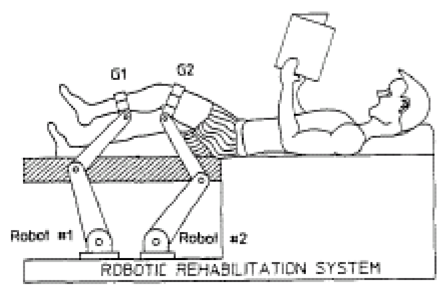


Figure 2-1: Simple human joint rehabilitation robot

The use of robotics to improve or even enable a patient to regain walking ability has solved some problems. With traditional therapy, multiple therapists are needed to work together in order to both support the patient and to guide the movements of both legs [15]. This is particularly intensive both in personnel and in cost. One robot, developed by Chicago PT, assists a single therapist in helping a patient regain walking ability [7]. This robot is adaptive and allows for more natural motions, which is an improvement over exoskeleton/treadmill robots developed previously. Many other gait-training rehabilitative robotics, including exoskeleton devices, can be found elsewhere [i.e. 15]. The focus here will be on upper extremity rehabilitation.

The use of robotics in arm rehabilitation has been under serious development for some time now. The weaker muscles and thus movements of the arm allow for less expensive, weaker motors to be used. Also, the balance of the patient is not a major concern as it is with lower limb rehabilitation. If torso support is needed, the patient can be easily strapped into a chair. Upper extremity rehabilitation may also be more beneficial for most users, since the use of arms is needed for simple necessary tasks such as eating and maintaining personal hygiene. For these and possibly other reasons upper arm rehabilitation is the first endeavor of many rehabilitation projects, including this one.

The earliest documented system developed for arm rehabilitation was the MIT-MANUS [30], of which a final version is shown in Figure 2-2. The development of this device began in 1989 to address the need for a low-impedance robot that was backdrivable, unlike all industrial robots at the time [32]. Impedance control [23] was implemented to achieve these means, especially that of safety. In this control scheme, the user moves the device, which reacts with a force when necessary. This system uses a Selective Compliant Articulated Robotic Arm (SCARA) to assist a user in planar movements. The patient grips the end-effector of this device and is led through exercises that have visual feedback on a computer screen, as well as the haptic (force) feedback of the serial manipulator. Newer iterations of this project allow for three-dimensional movements [29]. This device was used to prove that robotic therapy was at least as good as traditional therapy and that patients would accept it, though they prefer working with another human.

Figure 2-2: Two versions of the MIT-MANUS device

Two styles of the MIT-MANUS have been studied in several clinical trials. One such trial showed that users who truly used the robot as opposed to those who were just exposed to it and received traditional therapy actually gained more motor functionality in the upper arm – and continued to have better functionality over the non-robotic therapy group three years later [32]. The project has been published in many journals and research using this system is continuing in multiple centers as of the publication of this work.

Another class of physiotherapy robots are ones referred to as "mime" systems. The Mirror Image Movement Enhancer (MIME) device uses a patient's unimpaired arm to help perform therapy on the impaired arm [9]. Both arms are attached to splints which set the full position of the forearms. The system can give assistance with several types of therapy styles including passive and active modes. The unique feature of this system however, is the bimanual mode. In this mode, the user moves his or her non-impaired arm, which is attached to a serial-configured (master) manipulator that measures the position. This information is sent to a small industrial (slave) robot that mimics the position of the master. The slave robot, originally a PUMA-260, was limited in force to make the system safer. (More recently, a PUMA-560 was incorporated to allow for larger, stronger movements.) This master/slave configuration, shown in Figure 2-3, allows the user to truly direct their therapy and exercises. It also helps to maintain safety, since the system would only move in ways that mirror the other arm. The patient has control in determining how far or how quickly a movement is made and can immediately make adjustments should the impaired arm become uncomfortable. The problem with this system is that it requires that one arm be functional. While this may be the case for many stroke survivors, the system would not work in the bimanual mode for other patients including those with Multiple Sclerosis or spinal cord injuries that have two impaired arms. For home use, this system may be difficult to implement due to the cost and space requirements of a floor-mounted robotic arm, as well as possible safety considerations for nearby children and pets when no therapist present. Still, the availability of three-dimensional movements and full forearm positioning abilities are clear advantages to this system. Initial clinical trials have shown that this system lends some additional improvement in both strength and functionality of limbs over more traditional therapeutic methods.

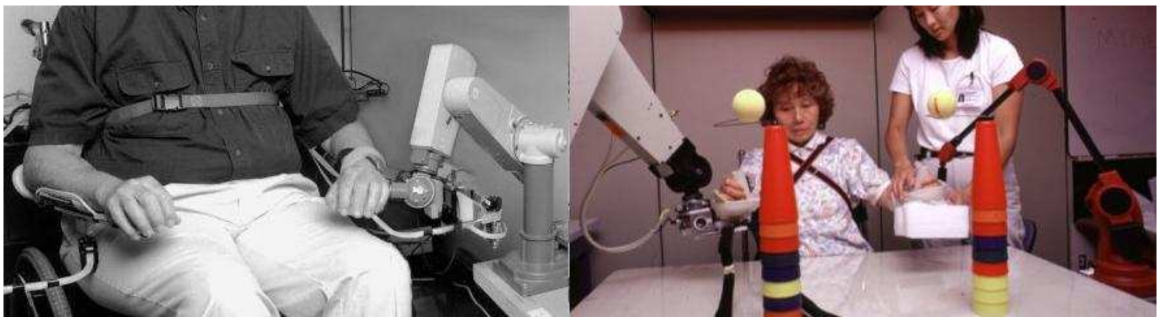


Figure 2-3: Two iterations of the MIME project

A research group in the United Kingdom is working toward a robot that can monitor and control the entire arm, instead of only the hand [11]. Rather than using impedance control as the MIT project, this design uses admittance control. This type of control requires the measurement of force in order to determine the appropriate trajectory. Due to the implementation of the sensors, this type of control system usually is more costly and also less compliant. The determination of the kinematics of the arm is also important in the precise performance of this system. More work is being done to further implement and test this device and the results of using such a control scheme could be interesting to further the range of the body that participates simultaneously during a therapy session.

Alternatively, a device that depends heavily upon impedance control is currently being investigated which uses pneumatic actuators rather than electric ones [47]. The use of newer pneumatic actuators allow for a range of positioning, but at a fraction of the cost of comparable electric motors for the given power. These actuators are also light, which could help to make a system more portable. However, accurate positioning of these actuators is not practical with force-based control systems due to factors such as fluid compression. To remedy this, a position-based impedance control scheme was implemented, thus negating the influence of the system dynamics.

Training may involve using multiple senses. While haptic (force) feedback to the user is implemented in many rehabilitation systems, one system developed at the University of California also implemented visual feedback [19]. This experiment used three methods. The first involved only visual training (watching the end-effector follow an ideal trajectory), the second involved haptic guidance only, while the third method combined both of the previous aspects (all shown in Figure 2-4). A study with this system showed that learning with vision was more effective than learning solely by learning a trajectory.

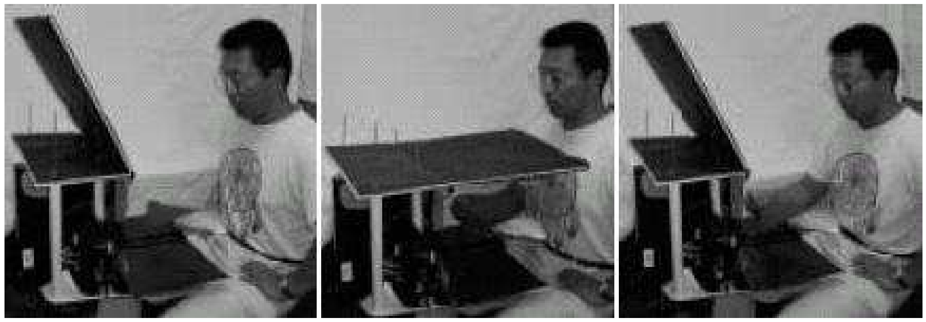


Figure 2-4: University of California haptic guidance experiment (Visual-only, Haptic-only, and Visual-Haptic configurations)

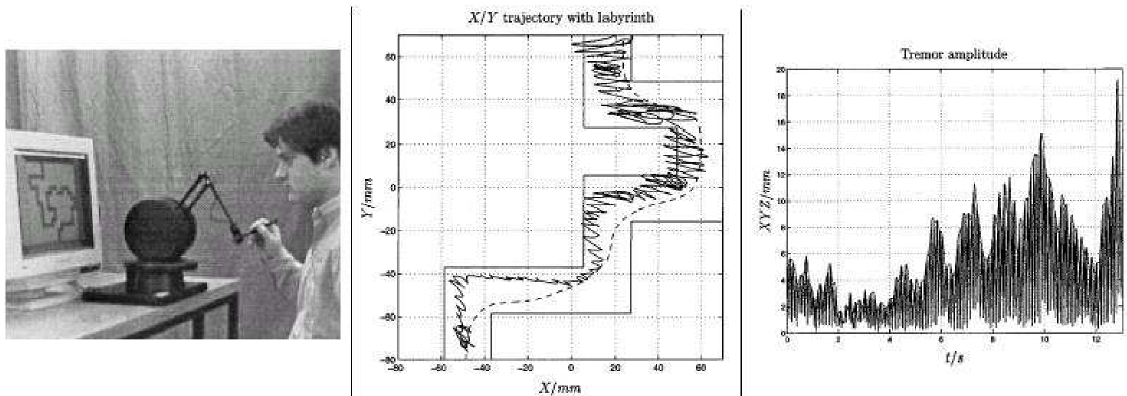
Obtaining feedback information about a patient's progress is another important issue in rehabilitation. Clear, verifiable measurements allow for better evaluation and treatment. Depending upon the conditions and the therapist, testing of limb ability usually include one or more of the following: dexterity and speed (of one hand as well as both together), writing ability, muscle strength and joint range of motion [1]. The measurements and interpretation of results can vary significantly between therapists [21]. The feedback can also be used to obtain knowledge about how the central nervous system functions and how recovery occurs in the human body [43]. Many rehabilitation systems have the ability to measure a patient's progress. For example, the MIT-MANUS robot was used in a study to map out what areas of the brain work during various types of physical therapy [28, 43]. However, there are some systems designed primarily for evaluation purposes.

An example of a device developed to quantitatively and accurately assess the motor impairment of arms is the Assisted Rehabilitation and Measurement

Guide (ARM Guide). The ARM Guide is a device that allows for only linear motion using a chain drive [46], as shown in Figure 2-5. This system is suited for reaching movements and can measure both the hand position and force generated. While designed primarily for measurements, this device is able to also assist in therapy for one degree-of-freedom movements. This device is currently being tested in clinical trials. Unlike the two previously mentioned systems, the clinical trial of this device does not as of yet show a significant improvement between using the robot or not [44]. The reason has not yet been determined. It could be that the assistance for this movement is not necessarily beneficial, or that the system does not provide better therapy than a therapist.

Figure 2-5: ARM Guide device

Another research group based in Slovenia devised a system for clinical evaluation using the commercially available PHANTOM haptic interface [1], as depicted in Figure 2-6a. The main concern of this project was to develop software for testing, rather than to design the physical device. Using different formats for presenting the results, information regarding the disability and its severity is obtained. These include trajectories (Figure 2-6b), reaction forces, tremor amplitudes (Figure 2-6c) and motion frequency.



(a) (b) (c)

Figure 2-6: University of Ljubljana haptic interface and two forms of output

In an effort to move these robots from laboratories and clinical labs to the home, some research is now focusing on creating telerehabilitation systems, particularly by using the Internet. One system, referred to as “Java Therapy” focuses on this approach [45]. The system, which uses a commercial haptic joystick (shown in Figure 2-7a) and some accessories, exploits web programming as a means for creating a therapeutic interface. This apparatus can also be used as a testing and communication platform. Because of this design, the cost is kept quite low. The therapists simply logs in to retrieve or send information pertaining to the patient’s exercises and progress. A more recent iteration of the system now uses a more extensive manipulator, pictured in

Figure 2-7b. This is has been dubbed the Training Wilmington Robotic Exoskeleton (T-WREX) and can measure or manipulate the position of the entire arm [46, 49].

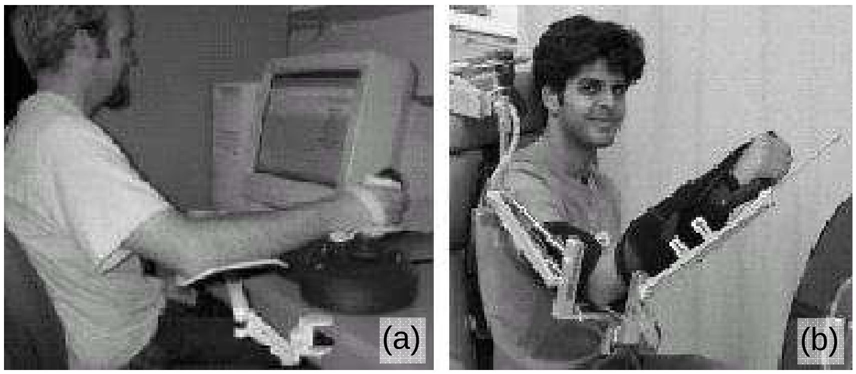


Figure 2-7: Two iterations of the T-WREX system

The Rutgers Arm offers another solution [33] to developing an internet-based system. This uses a virtual-reality environment and a platform that monitors and records arm movement, shown in Figure 2-8. The user simulates tasks such as pick-and-place while being monitored by the system and a remote therapist. While this system is extremely safe and light, there is no haptic feedback, which provides limited possibilities outside of assessment and general movements.

Figure 2-8: The Rutgers Arm

One study proposed that the accurate control performance of a rehabilitation robot is not very important and that safety should be the major issue [39]. This same paper suggested the use of artificial muscle materials to be used due to compliance and similar characteristics to human muscle. While this is one good solution, the current costs of these materials would make the apparatus somewhat costly. In addition, further research has not been reported which may be due to the fact that the use of this material was not currently feasible for a rehabilitation robot.

Several cable robots have been introduced to the area of rehabilitation [6, 35, 55]. These systems, as opposed to systems that use a serial robot, have the advantage of being lighter, easier to manufacture and repair, and allow for a larger range of motions. Cable systems in general, however, have limited speed ranges and can be somewhat inaccurate due to backlash and sensor positioning. There is also the problem of keeping the patient from interfering with cables, which offers the potential for injury or further positioning errors. A very nice feature is the built in safety for human-robot interactions incorporated by the compliant nature of certain cable mechanisms [52]. The advantages offer much promise and make it worthwhile to find solutions to the disadvantages.

2.2 Survey of Cable-Driven Robotics

Cable-driven robots have been under research for over two decades. Some systems include the Robotcrane, Charlotte, the Texas 9-string, SPIDAR, the 7-cable master, 8-cable haptic interface [58] and KinoWire [38]. Usage for cargo transportation is one of the most widely accepted applications [51].

These systems were developed in order to gain the advantage of light weight and larger ranges of motion compared to more rigid robots. This makes the system more energy efficient [22]. The flexibility also allows for easier construction and manipulation of the configuration, which is being exploited by some [4]. Some factors to be wary of include positioning the cables such that they do not interfere with each other as well as with surrounding objects. The ability to build a robot that is easy to design and low-cost are some key features of cable-driven systems, however [40].

One example of a cable-driven rehabilitative device is the Multi-Axis Cartesian-based Arm Rehabilitation Machine (MACARM) project [35]. The goal of this design was to create a rehabilitation tool that exploited the benefits of cable-drive, including that of being modular and reconfigurable by use of these modular parts. The software and control architecture is also very flexible. The

MACARM system allows movements to be bound by paths, user-applied forces or inputs such as joystick commands. The prototype consists of eight cables and eight “active modules” – the control unit for each cable, as shown in Figure 2-9. As of now, this is the only cable-drive system in literature that was developed specifically for rehabilitation from the very start.

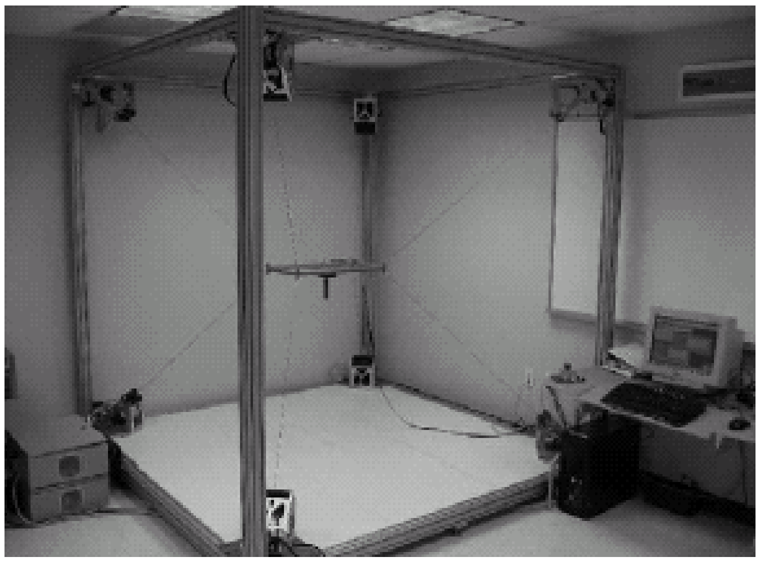


Figure 2-9: MACARM prototype

A system very similar to the MACARM project was developed in Germany at about the same time. This team explored several configurations from four to eight cables and possibilities for such a device, calling the project SEGESTA [6, 17, 22].

Another cable-driven device was developed at Ohio University almost a decade earlier than the MACARM, though not specifically for rehabilitation. This device was simply called a cable-suspended haptic interface [55]. The implemented device used four cables and was mounted in a table so that translational characteristics were the focus (see Figure 2-10). The project also proposed a six degree-of-freedom device, which is nearly identical to the MACARM project, though this was not reported to have been physically implemented. The main concern of this phase of the research was to determine mathematical models including the kinematics and control. Later, the focus of the research seemed to have shifted solely to planar cable robots.

Figure 2-10: Four-cable planar "cable-suspended haptic interface"

In later works by Ohio University and collaborators in Italy, two arrangements for planar robots were proposed [20, 57]. These two arrangements, now called planar cable-direct-driven robots, were chosen because both have necessary actuation redundancy, but are not redundant in kinematic respects. One was the four-cable system implemented previously, while the other was a three-cable system, both shown in Figure 2-11. Analysis was done on both arrangements to determine the kinematics, modeling and control. It was determined that the four-cable system required less energy [57], but that the three-cable system had smaller tracking error [20]. In addition, a later design permitted the four-cable system to allow some orientation of the hand to be determined. More studies were performed on the four-cable device with respect to better models and cable interference by this research group [56, 58].

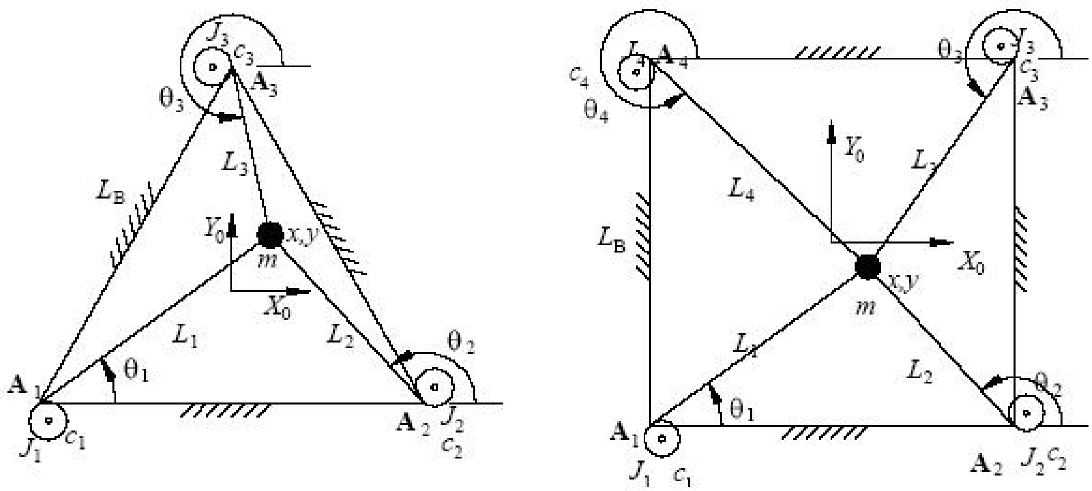


Figure 2-11: Two planar "cable-direct-driven robot" configurations

Three cables are the minimum required system for full force-feedback in all directions of a plane, though four cables allow for a larger work area as shown in the above figure. Using more than four cables is overly redundant for simple planar motions and is not considered. The three-cable device is similar to that which is developed in this thesis.

Overall, the use of a cable-drive design for a rehabilitation system seems to offer the best solution for making a practical home-based device. While there are advantages to the other systems, the disadvantages make them less suited for a more immediate release as marketable products. The cable-drive system is further detailed in the next chapter, in addition to solutions to some of the problems of this type of drive mechanism.

Chapter 3

SYSTEM DESIGN

While the various systems mentioned in Chapter 2 each have unique advantages and disadvantages, most do not meet many of the criteria outlined in the Introduction to be suitable for home-use. The systems require the constant presence of a therapist, are too expensive, are unattractive and/or are too bulky for a household setting. Overcoming these challenges would allow a system to be marketable and available for more wide-spread home and office use. This also allows the innovations of other systems to be adapted for inclusion in home systems.

The proposed system does not take into consideration the joint angles of the arm, but rather simply the position of the hand within a planar frame. This device is intended for use in regaining general large motions of the upper limbs. How the user positions his or her arm to attain the appropriate position is considered irrelevant for this particular project, though this is explored by other projects and studies [37].

3.1 Design Overview

The concept of using cables is a very promising step to take in rehabilitation robot research. The decision to use three-cables was made because it was the simplest fully-constrained layout for the purposes of defining the position of the end-effector (and thus joystick) on a plane *and* providing force (haptic) feedback in all directions on the plane. Two actuated cables and a passive tension device (such as a spring) could be used to define a position, but the ability to apply forces on the user would be limited in some directions and the force felt would be distorted throughout the work area by the varying spring tensions. The use of a four-cable device is the final viable option for planar motion without having too much redundancy. While the use of four cables may be more efficient in some respects [57], the reduction of one actuator with its related parts and control circuitry reduces the initial costs and long-term maintenance of the machine. Furthermore, the actual consumer electrical energy consumption of operating the device with only three motors is smaller since there is one less motor (and related circuitry) that must be powered. This all helps to lower the total operational cost of the device. The criteria for this system are based primarily on marketability factors, rather than ideal models or mathematical outcomes.

The planar layout allows for many useful exercises to take place, despite the limited range of motions. One project found that a linear or planar system was good enough for most therapy and that the full range of many robots was not always utilized, anyway [26]. The lower-cost and safety of the planar layout is sufficient for most users and has possibilities for a greater range of motion in various planes, as discussed in the final chapter.

The work platform consists of a frame on top of a table containing all of the working parts. The frame also supports an acrylic sheet surface, discussed in more detail later. Electrical motors are used for actuation of each cable and are mounted vertically underneath the table. The drive shafts protrude from the top of the table and each has a spool mounted to it. Each spool guides a cable within the groove and allows the cables to be wound and unwound, decreasing and increasing the length of the cables. This basic set-up is shown in Figure 3-1.

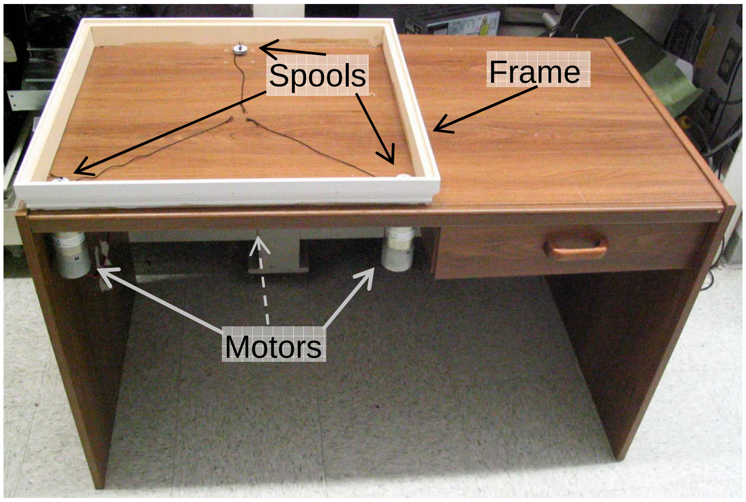


Figure 3-1: Basic layout

The robotic portion of the device is constructed with the three motors arranged in an equilateral triangle, as shown in the previous chapter as well as Figure 3-2a. A cable is mounted on each spool, attached to a motor shaft, and all three cables meet at a circular center platform (the end-effector), referred to as the “lower base.” The kinematic center is located where these three cables would intersect if they continued through the lower base. The handle the user grasps during operation of the device, referred to as the joystick in this work, is offset slightly above the kinematic center to allow space for the user to comfortably rest the working hand. The calculations for position are therefore centered at the center of the lower base and not the joystick. This poses no great problem since the general position of the arm is all that is of concern.

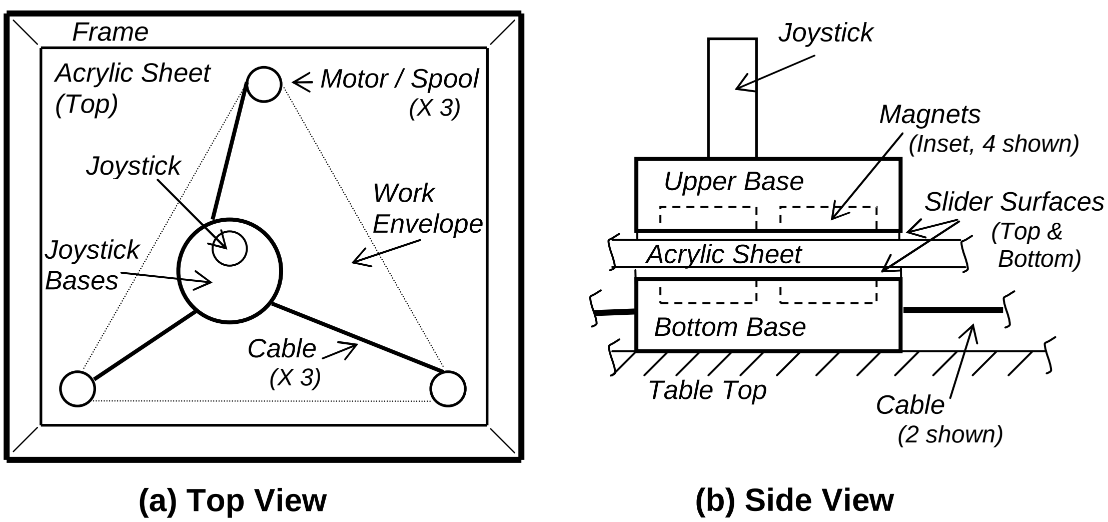


Figure 3-2: Detailed layout of work platform

To aid in ergonomics, safety and cleanliness, an acrylic sheet encasing is mounted over the work area (shown in both Figure 3-2a and 3-2b). This allows either arm to rest anywhere near the joystick without interfering with the cables or being harmed by the motors. It also helps to keep most dirt and debris out of the motors for maximized device life. The joystick is physically part of another circular base (referred to as the “upper base”). This upper base is magnetically connected to the lower base using strong magnets (Figures 3-2b and 5-3). While the magnets connect the bases securely enough to transmit haptic feedback, the joystick can break with the lower base in the case of excessive force. This feature makes it less likely that a user will incur injury should the bottom platform move too quickly and/or unexpectedly. In addition, there is no damage to the device when this break occurs. Another added benefit is that the device may still be used as a functional table with a flat surface when the upper base is removed. However, the tasks that could be performed without damage to the somewhat flexible acrylic layer would be limited and the presence of the strong magnets in the lower base (which is not as easy remove as the top base) would be a concern for nearby ferrous objects and magnetic media such as hard drives and credit cards. Both the upper and lower bases are as small as possible to allow for a larger workspace [18]. The bases were made large enough to accommodate the arrangement of magnets and also to give plenty of space for the user to rest his or her training hand comfortably.

The device incorporates visual cues to aid in the effectiveness of treatment. One study in particular showed that “… haptic guidance alone is not an optimal method for training for position or shape accuracy when direct vision is available during the task” [19]. The acrylic sheet table top allows for paper to be temporarily attached, attaining the ability to mark paths for each exercise by running the robot once first without the user interacting with the joystick. The user may then run the program again and use the joystick to follow this ideal path. It is even possible to mark the actual path he or she took by using a different-colored pen during their exercise. A penholder has been incorporated onto the joystick for this purpose.

A user interface is also provided by a LCD screen mounted into the table. This prompts the user for information such as what mode to put the device in as well as provides feedback on progress. A keypad of a few buttons is incorporated to allow the user to respond.

All motor controls and other electronics are mounted inside a desk drawer. This minimizes the chance of accidental damage to the electronics and keeps them out of sight for aesthetic purposes. In addition, the drawer can be locked so that the user or children can not access the circuitry and accidentally injure themselves or damage the unit.

3.2 Layout and Kinematics

An analysis of the kinematics of this device configuration has already been published by another research group [20, 57, 58]. This group explored a three-cable configuration and four-cable configuration as shown in Figure 2-11. They chose to focus their research on a four-cable configuration, however. Due to the decreased initial and operating costs as well as the higher reliability, a three-cable system was chosen for this project as the more practical choice for a lower-cost system.

Another possibility for determining the kinematics of this system uses the Anipodal Grasping Theory [13]. After review the literature and other sources [i.e. 3, 56], it was determined that this theory is not well suited for the particular case and arrangement of this cable-driven manipulator.

For this project, a very simple system of determining the kinematics was developed. While this is not the most mathematically eloquent, the kinematics are designed such that the calculations can be easily made using the device's control architecture, which consists of very simple microcontrollers with limited speed and calculation ability.

Work Area

The first step in deriving appropriate kinematics involves determining the actual work area of the end-effector, which in this case is the joystick base. In determining the work platform area, the distance between the centers of the motor shafts, L_{full} is first considered. This creates an area shaped like equilateral triangle as shown in Figure 3-3. However, to keep the lower platform from contacting the spools on each motor shaft, a smaller area must be considered for the base to move inside. Since the spools have an outer diameter of D_S, this is subtracted from each side of the outer triangle to create an inner work area as shown in grey in Figure 3-3. The area of the work platform is now an equilateral triangle with the measurement of each side represented by $L_{spool} = L_{full} - D_S$.

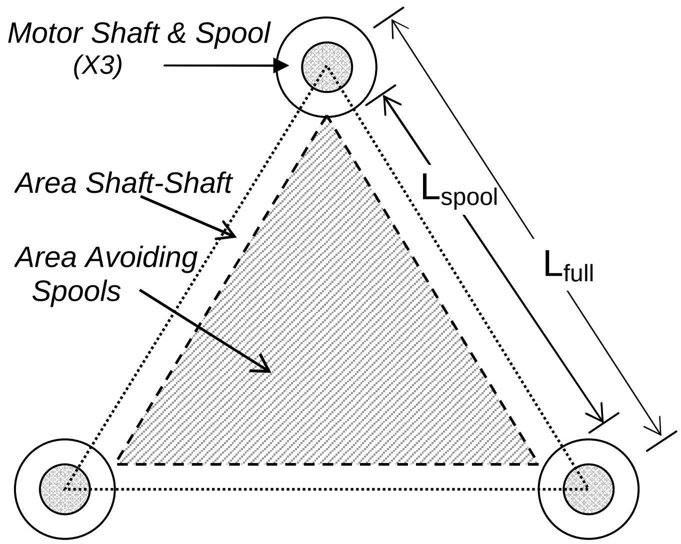


Figure 3-3: Allowable work area on table (no joystick consideration)

The area occupied by the joystick is considered next. As shown in Figure 3-4a, each cable requires an allowance for the spring (or small force sensor) to be mounted near the base. The lower base has a diameter of D_{LB}. In addition, an allowance must be made for each of the spring assemblies. To simplify the determination of the area occupied, the area is represented by the equilateral triangle shown in Figure 3-4a. For the purpose of all kinematic calculations, the movement of the manipulator will be calculated from the center of this base/spring triangle marked with a cross in the diagrams. The size and area of this triangle was calculated using the geometry shown in Figure 3-4b given the known lengths of the spring assembly and radius of the base. This resulted in a length of L_{LB} per side.

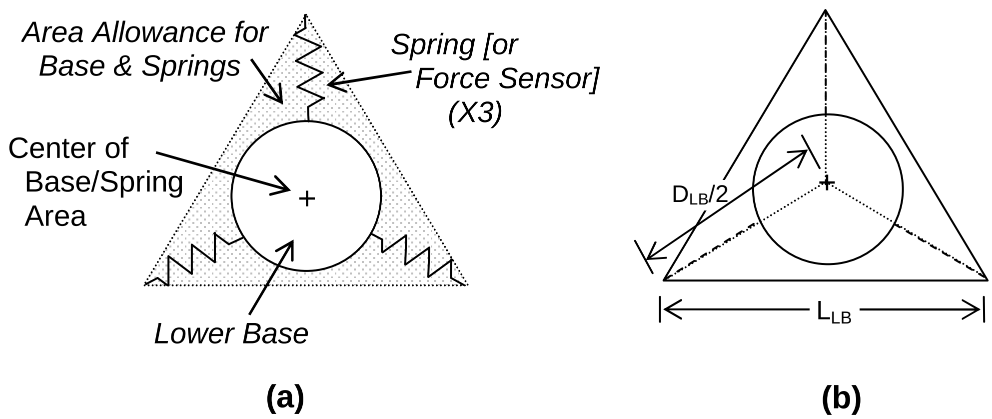


Figure 3-4: Area of lower base and spring assembly

To determine the actual work area of the device, the area that can be traveled by the center of the base/spring triangle in Figure 3-4 must be considered in the extremities of the work platform area found previously in Figure 3-3. Every side of the work platform area loses the length of one side of the base/spring triangle. The result is that each side of the work area is $L_{WA} = L_{spool} - L_{LB}$.

Kinematic Equations

The next step is to determine easily programmable kinematic equations for this device. The Cartesian coordinates of the position of the end-effector must be translated to the three cable lengths (and ultimately encoder positions) in order to facilitate the ability for the device to follow a pattern of movements previously programmed in the play and assist modes. This is the inverse kinematic situation. Also, the lengths of each of cable should be able to be correlated to the position of the end-effector, giving the forward kinematic solution for use in other the other two operational modes.

The kinematics are based upon the diagram shown in Figure 3-5. As shown, the *x*-axis will be always considered to be in the horizontal direction and the *y*-axis will be considered to be in the vertical direction for the duration of this thesis. Also, the kinematic origin of the work area will be considered in the lower-left corner.

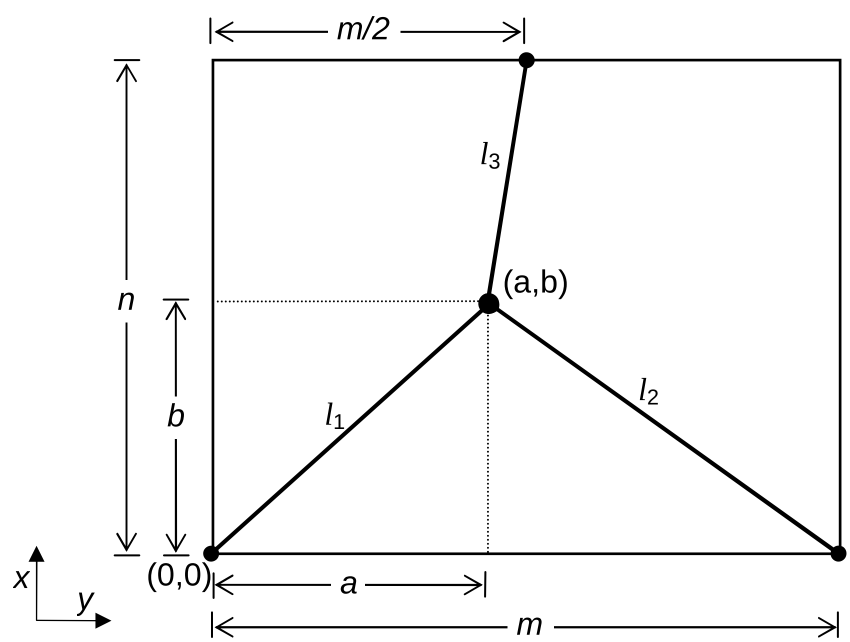


Figure 3-5: Kinematic layout of the work area

Using the diagram in Figure 3-5 and geometric relationships, the following equations for the inverse kinematic solution were found:

$$l_1 = \sqrt{b^2 + a^2} \qquad \text{(Equation 3-1)}$$

$$l_2 = \sqrt{b^2 + (m-a)^2} \qquad \text{(Equation 3-2)}$$

$$l_3 = \sqrt{(n-b)^2 + \left(\frac{m}{2} - a\right)^2} \qquad \text{(Equation 3-3)}$$

The three equations above were then used to determine the equations for the forward kinematic solution:

$$a = \sqrt{l_1^2 - b^2} \qquad \text{(Equation 3-4a)}$$

and $b = \sqrt{l_1^2 - a^2}$ (Equation 3-4b)

$$a = m - \sqrt{l_2^2 - b^2} \qquad \text{(Equation 3-5a)}$$

and $b = \sqrt{l_2^2 - (m-a)^2}$ (Equation 3-5b)

$a = \frac{m}{2} + \sqrt{l_3^2 - (n-b)^2}$, if $a \geq \frac{m}{2}$ (Equation 3-6a)

or $a = \frac{m}{2} - \sqrt{l_3^2 - (n-b)^2}$, if $a \leq \frac{m}{2}$ (Equation 3-6b)

and $b = n - \sqrt{l_3^2 - \left(\frac{m}{2} - a\right)^2}$ (Equation 3-6c)

As determined earlier, the area of the work space is such that for this project $m \equiv L_{WA} = L_{spool} - L_{LB}$ and therefore $n = \sqrt{(L_{WA})^2 - \left(\frac{L_{WA}}{2}\right)^2}$.

It should also be noted that the system of equations are over-constrained in both the forward and inverse cases. This is due to the fact that three actuators are being used for only two degrees-of-freedom. With tension needed on all cables at all times and to have the ability to apply force in any direction, however, the use of at least three actuators is essential. One advantage that can be exploited is that this allows for a check that all calculations made by the processor are correct. Should there be a discrepancy above a pre-determined threshold (to account for rounding errors), then the processor is able to signal an error that something is wrong.

Motor Movement Calculations

The final calculations that are needed before programming involve the determination of the rotation of each motor and the resulting linear motion. The diameter of the spool groove where the cable winds, D_{IS}, is known. The diameter of the cable is D_{cable}. The cable can wind within the spool three times before the cable begins to overlap a previous layer of cables. When a previous layer is overlapped, it is as if D_{IS} were increasing. This increase is equivalent to twice the value of D_{cable}. The equation for calculating the circumference of the winding on the spool (from the center of the cable), considering the layer would then be

$$C = 2\pi\left(\frac{1}{2}\cdot D_{IS} + [Layer]\cdot D_{cable}\right) \qquad \text{(Equation 3-7)}$$

as shown in Figure 3-6. To determine the approximate current layer, use

$$Layer = \frac{MaximumCableLength - CurrentPosition}{WindsPerLayer \times D_{IS}}. \qquad \text{(Equation 3-8)}$$

This circumference corresponds to the linear motion of the cable for every full rotation of the motor shaft.

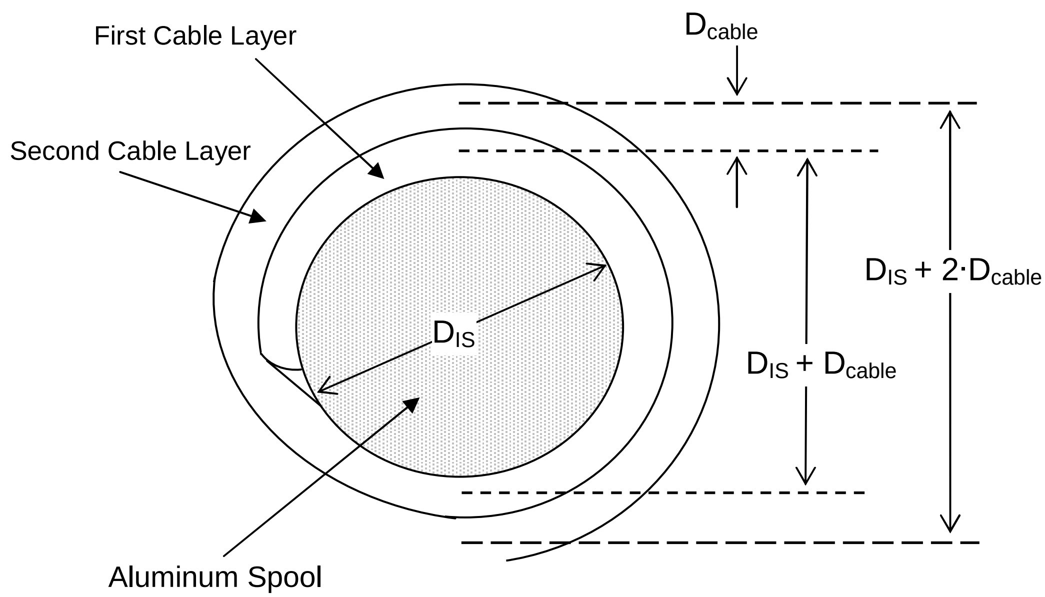


Figure 3-6: Spool thickness due to cable winding

Using information from the encoder, motor controller equation and gearing of the electric motors used, it is known that there are 5000 encoder counts per revolution, as seen from the motor controller. Dividing the linear motion by the encoder counts yields the distance traveled for every count of the encoder. This is used to determine the maximum distance that each cable may travel from the home spool of that motor and trigger an error if a motion requires a movement outside of this range.

3.3 Operation Modes

The control and operation of this robot requires that some assumptions be made at this point. Some minimum amount of strength and function of the user is assumed. The user must have enough strength to maintain a grip on the joystick (or have aid from a therapist in doing so). In addition, the user should be mentally capable of using this device and comprehending the instructions if it is being used without any supervision. Audio instructions and signals could be implemented for patients who are illiterate or who have poor vision. Larger buttons, LCD or voice recognition are options for those who may not be able to use the prototype configuration. If the user is extremely impaired in any other way, an unsupervised session using this system is not intended.

The device has the capacity for four modes of operation:

1. Play Mode
2. Assisted Mode
3. Assessment Mode
4. Record Mode

Only the first case, the Play Mode, is fully implemented within the scope of this thesis. In this mode, programs that were pre-loaded into the device via a personal computer are selected and run through the user interface. The device simply runs through the programmed motion without regard to forces applied by the user. No data about the user's resistance is recorded and the user is expected to maintain full contact with the joystick, but to not try to force it in any movements. The user acts mostly passively in this case. This mode serves as a method to practice the motion repeatedly without straying from the exact path. This simulates the “repetitions” of a given movement that the therapist takes the user through during regular physical therapy sessions.

Further development of this platform allows for the other three modes of operation such as the Assisted Mode. In this mode, the user actively moves the joystick and the device senses the motion and applies forces to give full haptic feedback. The motors change speed, direction or come to a stop to allow for this haptic feedback since it causes the user to feel a given range of forces when moving the joystick. The haptic feedback can be used to provide increased resistance force as the user strays further from the desired path and/or to provide resistance during the regular motions in order to simulate an increase in friction or contact with an object. This use of the haptic feedback can also be used with a specially marked paper attached to the acrylic work surface to further aid in the realistic feel of an exercise. In a highly developed form of this device, this mode could be used in conjunction with a virtual reality system to provide visual and audio accompaniment to the forces felt.

The third option is an Assessment Mode where the user moves the joystick and the robot moves only to precisely follow and record the motion. The user feels no resistance, making the device seem passive though it is moving along the user’s trajectory. During this time, data is recorded that can be analyzed to score the user’s success in completing the exercise properly. The speed, direction, force and position of the movements can be analyzed and the therapist can use this data in the way that is the most meaningful the particular patient and therapy goals. This allows for a quantitative assessment of the user's performance, rather than the therapist needing to rely completely on his or her judgment.

The final mode is a Record Mode where the device operates much like in the Assessment Mode, but the trajectory information is recorded in a format to be a model and played back in any of the other three modes. This is beneficial for a therapist to use as it is expected that they will be uncomfortable programming via a personal computer, which requires either a list of all intermediate points, a drawn trajectory or equations for the desired movement including relevant force and speed data. This mode has no real use to the user unless he or she would like to record his or her motions (without haptic feedback) and then to watch the device move in the Play Mode to see the motions in action.

Backdriveability

One problem inherent in the control of this type of system is that each cable goes slack during some point in the operation, especially during fast movements [20]. While more complex control schemes are one solution [57], this device uses a suitable control architecture as well as the very simple solution of using a slightly compliant cable to allow for some errors during motion. Though this creates some inaccuracies, the benefit of a softer feel and additional safety for the user against sudden motions is gained. The goal of this project is to create an inexpensive and functional device for generally large motions of the arms and this has been kept in mind. Allowing these small inaccuracies significantly decreases construction and controller costs and keeps the device within the stated goals of this project. Depending upon the scale and use of the system, the stiffness of the cables (or springs mounted in-line with the cables) can be adjusted to reduce or increase the free-play.

For all but the Play Mode of operation, the system must be backdriveable. This means that when the user applies force on the joystick, the system can move easily showing little or no resistance. This feature is essential for the measurement of progress in the assessment mode configuration as well the appearance of no force if the user is correctly performing a task in a therapeutic configuration. With the cable-driven design, the backdriveablity is usually obtained via a force sensor, though a position sensor on the end-effector could be used for slow motions if there is enough flexibility in the cables to handle a slightly slower response. The slight elasticity of the cables aids in the ability of this system to be fully backdriveable by giving the motors extra time to respond.

Impedance control is a cornerstone of many rehabilitation robots including the MIT-MANUS project [23, 24, 25]. In this arrangement, the user moves the device and it reacts with a force when needed. This allows for the device to be safer and lighter since it allows passive movements. It also allows for smaller, more inexpensive designs to be used than with other control schemes. Another advantage is that a “position based impedance controller does not require consideration of the system dynamics" [47]. This simplifies controller design significantly.

The robot as fully completed for this thesis uses position measurements as the only means for feedback. These are taken from the encoders of the motors when the device is guiding the user in the Play Mode. Measurements of force or position variations (due to the user's pressure) on the end-effector are critical for the other three operation modes. With the addition of force sensors, a comparison can be made between the current position (determined from the encoders) and the applied trajectory of the user (obtained from the force sensors). Depending on the operation mode, this may cause the command for the motors to change (positively or negatively) or to maintain the motor command in order to provide haptic feedback or the appearance of no force. This information could also be used to detect an error and shut down the device.

Operation Procedures

In all modes, the master controller monitors the progress of the user and the overall operation of the system. It sends commands for motor adjustment as needed. It also sends information to the LCD and handles requests from the user between exercise operations.

To maintain safety and ensure that users are less anxious about using the robot with little or no supervision, a set procedure must be completed during start-up. The device is powered-up simply with a switch, but the user is then prompted and must enter information to select what the device will do. This could also involve entering a password, should the user or therapist desire this option. The password will not only reduce the likelihood of unintentional activation, but will also allow multiple users to store data concerning personal progress and routines within the memory. After the device has been initiated for a given exercise in any mode, the user must press and hold the activation button on the joystick. Since the intended user is assumed to have limited muscle strength, the button requires little pressure. The activation button must remain pressed for the duration of the exercise in order for motion to continue. Before movement begins, the LCD, LEDs and a speaker all provide notification that the movement will commence. Should the activation button be released during an exercise, the device enters an emergency stop mode and motor power is disengaged. This is very similar to a “dead-man switch” found in many other devices. There is consideration for a shifting, shaking or slipping grip that is not an indication of a problem. If the user loses and regains grip of the joystick within a few hundred milliseconds of the loss, automatic continuation occurs without a full power-down.

The user must either complete an exercise or enter an emergency stop condition in order to communicate with the master controller through the keypad after an exercise operation has begun. This will insure that the user is never distracted needing to give the controller commands while the manipulator is in operation. The stop button on the user interface will cause an emergency stop, if pressed, however.

The emergency stop can be activated in three ways. It has already been mentioned that releasing the joystick for too long during normal operation will cause an emergency stop. This shuts down the power at the motor controllers and amplifiers. The device will also cause itself to go into an emergency stop if it detects a serious error, such as sensing the lower base hitting a cable spool or a sudden loss of data. Finally, there is an emergency stop button mounted prominently near the work area that will immediately disconnect the main power when pressed. For all but the release of the joystick, resetting after an emergency stop may require that the device first be realigned or repaired. After all errors are corrected, the device will again implement a full start-up procedure.

Shut-down is a simple procedure. If an exercise is completed, the motors automatically power-down, the speaker sounds and the LCD provides feedback and prompts for next task. This is just to let the user know that the device is stopped and they have completed an exercise. The user may opt to perform a device shut-down here, repeat the exercise or go on to another activity. An automatic-shut-down may occur after several minutes of no activity so the device is not active too long while unoccupied and unsupervised. If the main power switch is turned off, the device will power down immediately. This may cause information from the previous use to be lost and a realignment may be necessary to proceed after returning power if a task was in operation at the time of the shut down.

Errors during operation are handled automatically when possible. When this is not possible, a message on the LCD should appear. Depending on the error, the user may be able to make changes to repair the issue, such as performing a reset to correct a non-responsive controller. More serious errors would have to be handled by the therapist or service technician.

Chapter 4

CONTROL

The process of developing algorithms for control and implementing them are discussed in this chapter. Different arrangements and considerations are needed for the various modes and conditions that the robot is used in. The control of the device can be implemented using just the position or a combination of position and force. In addition, the user's ability to direct the system allows for some kinematic problems that must be solved. The theory and implementation of the control system is presented.

4.1 Position Control

The basic control loop for the Play Mode is fairly simple. Here, no feedback from the user is considered so only the motor position and desired position are incorporated. The overall diagram of the control system needed for this function is shown in Figure 4-1.

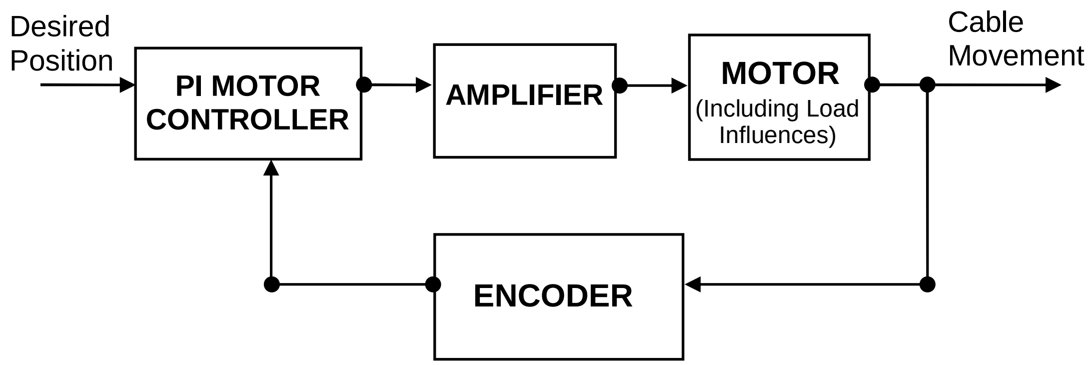


Figure 4-1: Overall control scheme in Play Mode

The control scheme shown above is typical for nearly any system using only position feedback. The desired position is determined and entered into the motor controller chip, along with any desired velocity information. The controller chip is itself a small control system using the position information from the encoder to generate a pulse-width-modulated (PWM) output signal to the amplifier and thus motor.

During the Play Mode, the system operates by defining intermediary positions (whether via real-time calculations or with a table of positions) and moving to each one. This means a desired position is entered and then the system moves toward the new position and signals when it has arrived so that the next position can be commanded. This process continues for every intermediary point until the entire programmed path has been followed. The speed of the end-effector during each motion can be altered depending upon the capabilities of the user.

The motor controller is a key part and takes care of many of the calculations needed for proper operation. It was quickly determined that implementing a chip to perform the necessary control loop functions would not be worth the lower cost. This is because the calculations can become intensive and easily create too much of a control loop delay. Instead, it was decided to use a general-purpose low-cost control chip available on the market. An Avago (formerly Hewlett-Packard) HCTL-1100 was selected to perform the control loop functions and be the motor controller as shown in Figure 4-1.

To initialize this chip, the desired mode type, velocity, acceleration and digital filter are entered. To operate any command only the desired positions (and velocity, if desired) need to be inputted. The controller then computes the necessary velocity and implements the desired control profile (i.e. position, integral velocity, proportional velocity or trapezoidal velocity control) in order to achieve the correct motor movements. The exact control mechanism it uses is not completely disclosed by the manufacturer, though it seems that it can approximate a PI control and several other control profiles. The digital filter used by this chip on the output signal is the only “tuning” permitted by the user as the chip does not allow for gains calculated by traditional control theory to be used.

In order to tune this motor controller, the first thing that must be determined is the sampling time of the HCTL-1100. This is calculated as $16(R0FH+1)\left(\frac{1}{f_{clock}}\right)$ where *R0FH* is the value the user writes in the chip’s Sample

Time Register (a decimal value between 15 and 255, a higher value increasing the sample frequency) and f_{clock} is the frequency of the clock used to operate the chip. In this case, the maximum allowable sample time of 255 and a clock of 500-kHz is used. This gives a sample time of 8.192 milliseconds. The chip uses a digital filter to allow for compensation to improve the desired response. The filter consists of a gain, a pole and a zero. A method for selecting these values is provided in the chip documentation (Application Note 1032) and was followed as described below to determine the theoretical values for the system.

The open-loop transfer function of the system, depicted in Figure 4-2, is found by modeling each component in the system in the discrete frequency domain.

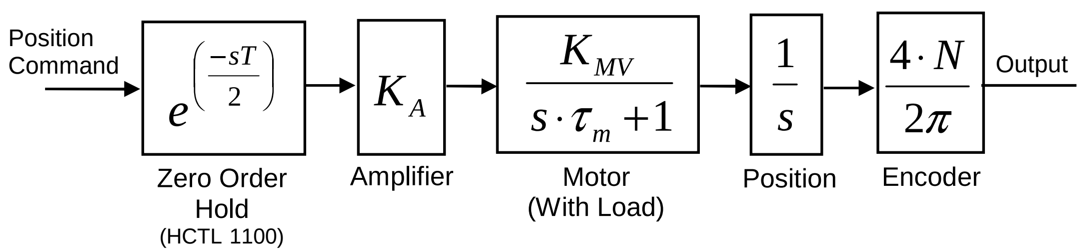


Figure 4-2: Open loop transfer function diagram of the system

The digital lead compensation filter internal to the chip is represented by

$$D(z) = \frac{K(z - A)}{z + B} \qquad \text{(Equation 4-1)}$$

where K is the gain, A is the pole and B is the zero. The filter is implemented directly before the zero-order-hold in Figure 4-2. In order to determine appropriate values for the filter, the rest of the system shown in the figure must be modeled. This will be done by looking at each component and determining the appropriate values.

The first block that is modeled is the zero-order-hold that is found inside the chip. It is only influenced by the system time constant, and is represented by

$$Z(s)_{zoh} = e^{\frac{-sT}{2}} \qquad \text{(Equation 4-1)}$$

where *T* is the sample time calculated previously.

The next component encountered is the amplifier. Since the amplifier uses pulse-width-modulation as an input, the gain is to be calculated as a voltage source amplifier by the equation

$$K_A = \frac{[MaximumOutputVoltage]-[MinumumOutputVoltage]}{[DutyCycleForMaxOutput]-[DutyCycleForMinOutput]}$$

(Equation 4-2a)

and can be reduced to

$$K_A = \frac{[MaximumOutputVoltage]}{[DutyCycleForMaxOutput]} \qquad \text{(Equation 4-2b)}$$

since both the minimum output voltage and the corresponding duty cycle are both zero. The output voltage is 12 volts and the duty cycle for the maximum output is 100%. This leaves an approximate amplifier gain of 0.12 when these values are evaluated in Equation 4-2b.

The motor model (with load considerations) was determined experimentally since the available specification sheet lacked the necessary information. The mechanical motor constant, τ_m, is one of two values that is necessary for calculating the values of the digital filter. This is assuming that the electrical motor constant can be neglected. The mechanical time constant was found by tracking the step-response. This involved starting the motor from rest and tracking the increase in speed over time by counting encoder pulses with a microcontroller over regular intervals. The motor achieved 63.2% of the full velocity under normal load in about 1.5 milliseconds, which defines the mechanical time constant. The load is considered to be constant since during operation the cable tension ideally remains constant. The testing was done with tension on the cable equal to that desired during operation of the device.

The voltage gain factor, K_{MV}, was found by including the gearbox and dividing the shaft speed in radians per second by the voltage applied to achieve this speed. Since applying 12-volts causes a shaft speed of 137 revolutions-per-minute, the resulting output is 1.1956 radians-per-second. The motor model, then becomes

$$G(s) = \frac{K_{MV}}{\tau_m s + 1} = \frac{1.1956}{1.5s + 1}. \qquad \text{(Equation 4-3)}$$

Finally, the encoder is considered in the model. This is simply the value of four times the counts-per-revolution of the encoder converted to radians. The encoder model is therefore

$$E = \frac{4N}{2\pi} = \frac{5000}{2\pi} \qquad \text{(Equation 4-4)}$$

since there are 1250 encoder counts (N) per revolution with the gearing.

The transfer function is then determined by multiplying all of the components together resulting in

$$TF_{OpenLoop} = Z(s)_{zoh} \cdot K_A \cdot G(s) \cdot E \text{ .} \qquad \text{(Equation 4-5)}$$

To facilitate the use of supplied graphs for filter design, the bode plot is required. The magnitude of the open-loop system model transfer function (converted into db) is plotted using

$$M(\omega) = 20 \cdot \log\left(\frac{K_{MV} \cdot K_A \cdot E}{\omega\sqrt{(|\omega| \cdot \tau_m)^2 + 1}} \right) = 20 \cdot \log\left(\frac{1.1956 \cdot 0.12 \cdot \frac{2500}{\pi}}{\omega\sqrt{(1.5\omega)^2 + 1}} \right)$$

(Equation 4-6)

and shown in Figure 4-3a. The corresponding phase of the model (converted into degrees) was plotted using

$$P(\omega) = \left(-\arctan(\tau_m \cdot \omega) - \frac{\pi}{2} - \frac{T \cdot \omega}{2} \right) \cdot (57.296) = \left(-\arctan(1.5\omega) - \frac{\pi}{2} - \frac{8.193 \times 10^{-3}\omega}{2} \right)(57.296)$$

(Equation 4-7)

and can be found in Figure 4-3b.

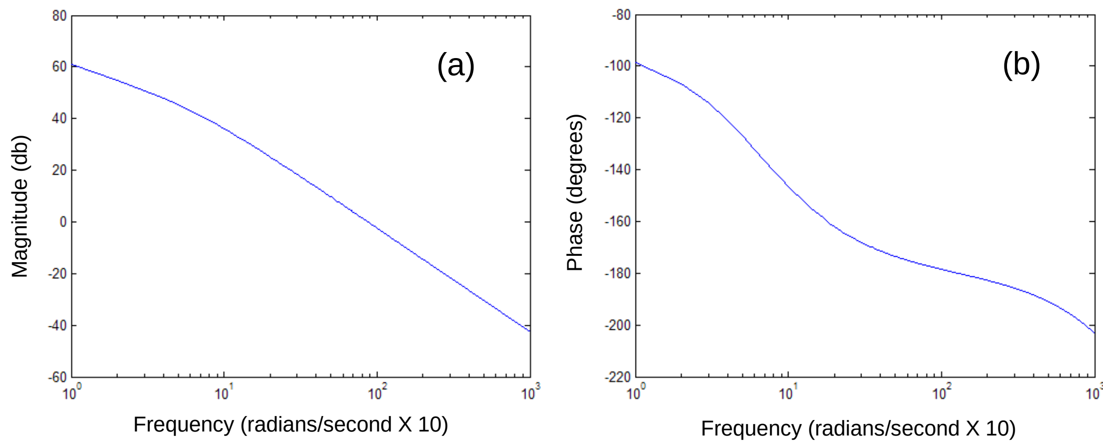


Figure 4-3: Bode plots of the uncompensated open-loop system showing (a) magnitude and (b) phase

The resulting phase crossover frequency is found to be 12.9-radians/second. The gain crossover frequency is 8.71-radians/second. Since the desired crossover gain should be the same as the desired bandwidth, this clearly exceeds the needed bandwidth of approximately 6-radians/second. Using the filter gain formula, which is the reciprocal of the system magnitude at the desired bandwidth, the controller manufacturer suggests a gain selection corresponding to

$$K_f = \frac{1}{M\left(\omega_{desired}\right)} = \frac{1}{6.45} = 0.16 . \qquad \text{(Equation 4-8)}$$

The datasheet also recommends that no poll or zero values are helpful since both parameters will increase the system response. Since the system response seems to be too fast according to the model, any additional increase is unnecessary.

This information was then used to set the filter parameters of the system. In practically implementing the motor control filter, a minimum gain value of one is required since only integers can be used. This is more than desired, but the system is able to function fine with no noticeable oscillation on the cables. It was not possible to increase the required gain since the sample time (which could be modified by lowering the value of the Sample Time Register or a slower clock rate) has no effect on the magnitude. As further verification that the values derived above are correct, a gain value of two as well as various values for the pole and zero values were tested. Any inclusion of a pole and/or zero or a gain greater than one guaranteed very noticeable oscillations and instability. During regular operation of the device, very low velocity commands are necessary to avoid the motor moving at full speed. The system does not respond noticeably faster with a velocity greater than about 75% of the maximum velocity value.

With the controller filter implemented, the motor control chip can then be used to operate the system. Upon looking at the various modes the chip offered, it was decided that the trapezoidal velocity profiling with point-to-point moves was the best option. This means that each intermediary point in the path is reached via a velocity profile that increases to the maximum specified speed via a specified acceleration, runs at the maximum velocity for some time and then slows at the same rate of acceleration when approaching the next point. The motor is run with this trapezoidal profiling to make the transitions between positions more smooth and easy on the user, rather than an instant start-up and sudden stop at each point. Due to this mode of operation, the controller operates as a simulated PI-controller. The derivative component would not be available for this type of control profile because it would undermine the desired output.

It is possible to use a different control profile, such as regular position control or a velocity control. The velocity control doesn't make much sense, at least in the Play Mode, since the positions are of a greater importance. The regular position control may apply a PID-style control algorithm with the proper digital filter values. The small gain in positioning accuracy was not determined to be as valuable as a smooth response as the device moved the end-effector from one intermediary position to another. In addition, since the overall position and rehabilitation of an arm is the concern, an error of several millimeters has no significant effect in the goal of the device.

It was also considered that during the movements to each intermediary points, different velocities for each cable would be needed to ensure that they were at the exact same point at all times between the specified position points. It was determined, however, that incorporating this was unnecessary. First of all, it would add significant time needed for the position calculations if the intermediary points were calculated in real-time. If the points are calculated prior to running and then loaded into the device as a list, the velocity table would take up memory that could otherwise be used for storing more programs or recording data. Secondly, the error during the movements can not be greater than the step size used between intermediary points. Since this is never greater than about five-millimeters, the error can never be larger than this since all cables are aligned at each intermediary point. The cables (and springs) are able to handle this small amount of error and would not be noticeable to a user or to the outcome of rehabilitation. For these reasons, velocity differences is not a factor considered during the Play Mode operation of the device.

This completes the control considerations for the simplest form of the system. However, the use of the other (i.e. Assist, Assessment and Record) modes of the device require a slightly more complex control scheme. This will be considered in the next section.

4.2 Force Control

To obtain control over the forces the user experiences, sensors are used to obtain feedback of the user's movements. This allows the device to either seem passive (though it is actually actively keeping all cables in tension) in order to record the user's unaided progress, or to provide haptic feedback to the user in order to simulate virtual objects or provide corrective feedback on movements. The sensor information is handled by the microcontrollers. The controller monitors the force information from the user and communicates any necessary changes to the motor controllers.

Assist Mode

The assist mode requires that the device provide varying degrees of resistance to movement as the user moves throughout the work area and/or over time. The forces could be exerted such that the user following the path exactly feels no resistance, but that some resistance is felt if the desired path is deviated from. The resistance felt could then be increased the further the user deviates from the ideal path. If desired, the resistance could also increase over the desired path itself in order to either simulate the moving of objects or to gently provide some speed control on the user as the path is followed.

This mode of operation requires a general control frame provided in Figure 4-4 below.

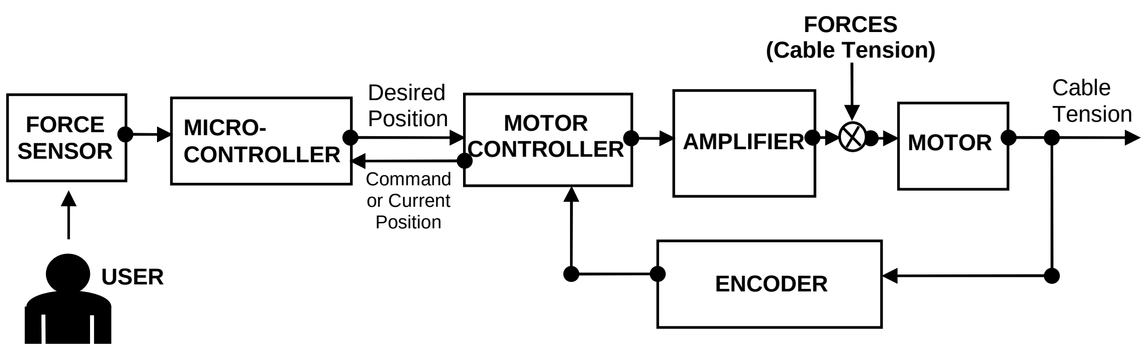


Figure 4-4: Overall control scheme in Assist Mode

In this scenario, the user initiates the movement and causes the cable tensions to change. This is picked up by the force sensors, which are being read by the microcontrollers. The microcontrollers then calculate the appropriate trajectory for each motor based upon the force to be felt by the user. This trajectory is communicated to the motor controllers to control their amplifiers and motors. The encoder data is monitored by the motor controllers which can make adjustments or send the current position back to the microcontrollers should the information prove useful in calculating the next trajectory. Overall, the bulk of the control system functions much like in the Play Mode and can use the same motor controller configuration. The main difference is that a force sensor is used to determine the user's movements and this is interpreted by the microcontroller so that position *and* speed commands can be altered to adjust the perceived force.

Another factor to be considered for the Assist Mode is that the motor load is a changing factor. This is because the user has much influence on the cable movement by applying forces. This can have the effect of decreasing or increasing the motor response, depending upon the direction of the force applied. It is therefore the job of the force sensor to quickly relay the user's influence to the microcontroller so that adjustments can be made to maintain proper tension in all cables.

In order to simulate a reflected force, the cable tension must increase for two cables as the user works against them. As a result, the tension is reduced and some slackness results in the cable not in tension, as shown in the example of Figure 4-5. This slack must never be permitted to become large enough to cause the cables to sag and slip out of the spools. A minimum tension must be actively regained via motor control. This minimum tension must be as small as possible while still being measurable by the force sensors that are employed.

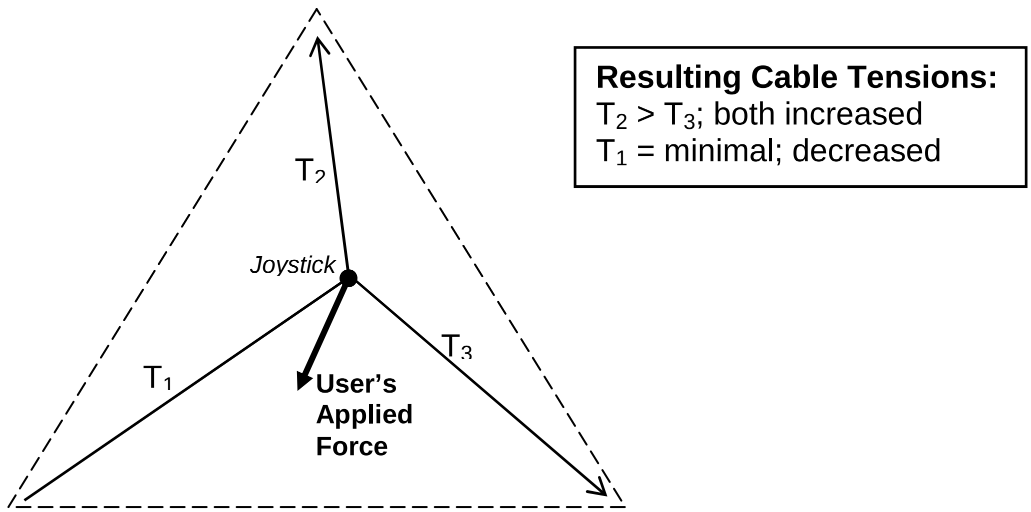


Figure 4-5: Example of cable tension due to user force

To be comprehensive, it should be noted that it is possible for one cable to have an increase in tension while the other two become more slack. This occurs when the applied force is directly in line with one of the cables, as in the example shown in Figure 4-6. It will be assumed from this point on that this case is possible and included in the discussion, though only the case with two taunt cables and one slack will be specifically mentioned.

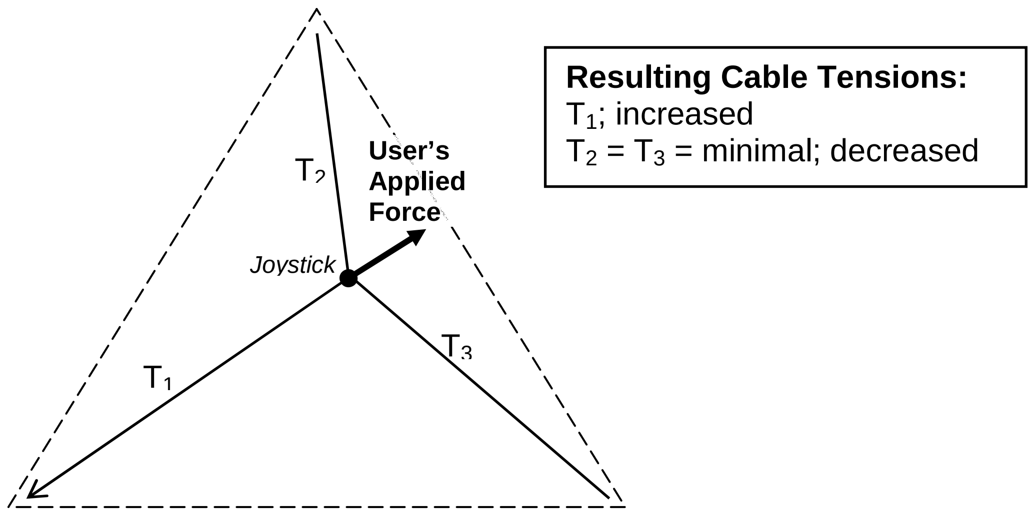


Figure 4-6: Example of cable tension due to user force in the same direction as a cable

To implement an increase in felt force while the user moves in a given direction, the speed of the end-effector should be slowed, stopped or reversed depending upon the desired force intensity to be felt. In the opposite condition, the motors should be run at a faster speed and in the direction of the user's movements in order to decrease the perceived force. Either way, it is a matter of correlating the direction and force of movements of the robot to that of the user in the appropriate manner.

The previous two illustrations show how the user applying a force causes the cable tensions to react. The robot, however, must use the tension measurements of the cables to determine the force of the user. This information is necessary to determine what the appropriate reaction force should be.

Both the direction and force of the user's movements are attainable through force sensors. Assuming one force sensor on each cable, the direction and magnitude of the user's force is found by considering the forces on all three cables. Defining tension as a positive force and also that an increase in tension causes a greater force, it is then never possible to have a negative (compression) force. It can therefore be assumed that the values of the force sensors will always be positive and will increase linearly with the tension. Given that there is an applied force, one cable will be under a minimally-maintained tension while the other two will have a greater amount of tension. In determining the force, the cable with the minimal tension should be ignored. This can be accomplished by eliminating tension values below a certain threshold from the force calculations. The magnitude and direction of the applied force can then be computed by adding the vectors of the other two cables with their sensor values since their lengths and tensions are known. An example is depicted in Figure 4-7 and detailed below.

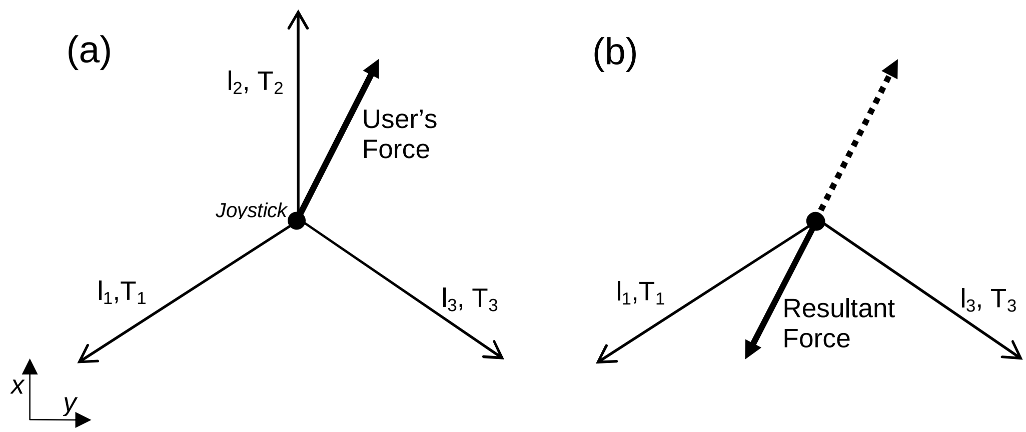


Figure 4-7: Example of an applied force on the manipulator

Given the example in Figure 4-7a above, it can be seen that the second cable will have only the minimum required tension. The majority of the tension will be on the first cable with a share on the third cable as well. By neglecting the second cable, as shown in Figure 4-7b, a resultant force can be calculated considering the vectors made by the first and third cable. This resultant force mirrors the direction of the force applied by the user, but is in the direction the user will feel the force since the joystick will press back.

The force and direction can then be used by the motor controllers to determine the new force that should be felt given the current position of the user. This information is then used to set new tensions in all cables.

Since the load on the motor impacts the HCTL-1100 motor controller filter values calculated previously, this needs to be briefly reviewed for this mode of operation. The impact in the control aspect is in the motor model due to load differences. Since the only significant change to the motor model is an increase in tension (resistive force), the motor response is slowed. This would increase the value of the mechanical motor constant when the cable is under higher tension. The cable tension is not productive if it becomes larger than the force holding the two magnetic bases together since the joystick will breakaway when this point is reached. As a result, the maximum tension is fairly limited and the increase in the motor constant would not be extremely significant. However, it will cause the entire system response to slow to some degree. As determined before, this is actually desirable and improves the system response with the correct speed. Therefore, no adjustments in the HCTL-1100 values are necessary to obtain a better control.

The speed of the motors play a larger role in this mode compared to the Play Mode since the velocity has a direct impact on the force felt by the user during an exercise. The trapezoidal profile mode of the motor controller should be used, but with the acceleration set to the full value such that the velocity values are attained as quickly as possible. This is because the trapezoidal profile mode is the only mode available that allows for both position and speed considerations simultaneously.

Assessment and Record Modes

From the perspective of the control considerations, the Assessment and Record Modes are identical. The only difference may be in how the data is recorded and formatted by the microcontroller since the data collected is for a different purpose. The general control diagram for these modes is shown in Figure 4-8.

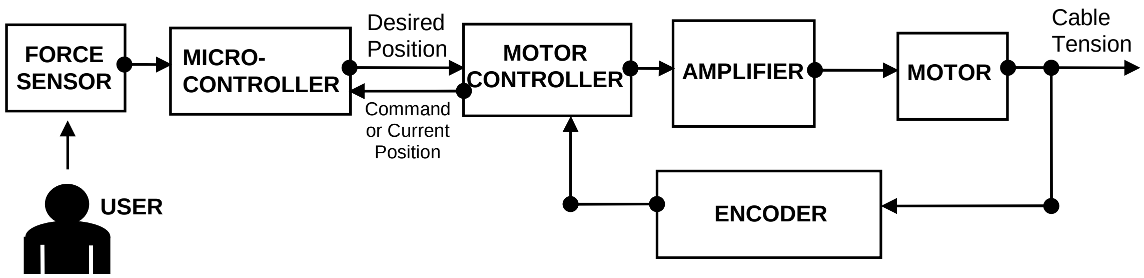


Figure 4-8: Overall control scheme in Assessment and Record Modes

The control in this mode is actually a bit simpler than that for the Assist Mode in that the load on the motors should remain fairly consistent at all times. Since the device should be precisely tracking the user and not causing any force to be felt, the standard cable tension is desired at all times in all cables.

This mode functions by the force sensors picking up tension differences. As shown before, a user-applied force causes two cables to gain tension while the third cable becomes slack. The microcontrollers calculate the trajectory of the user as in the example shown in Figure 4-7 for the Assist Mode and determine the direction and force the user is applying. This is then used to generate motor commands to move the cables into the position the user is requesting and to record these commands for later analysis or playback.

4.3 Singularities

Without the user's interaction (such as in the Play Mode), the system may not obtain any points in the area outside of the triangular work area presented in Figure 3-3 since tension on all cables can not be maintained. This is demonstrated in Figure 4-9 below.

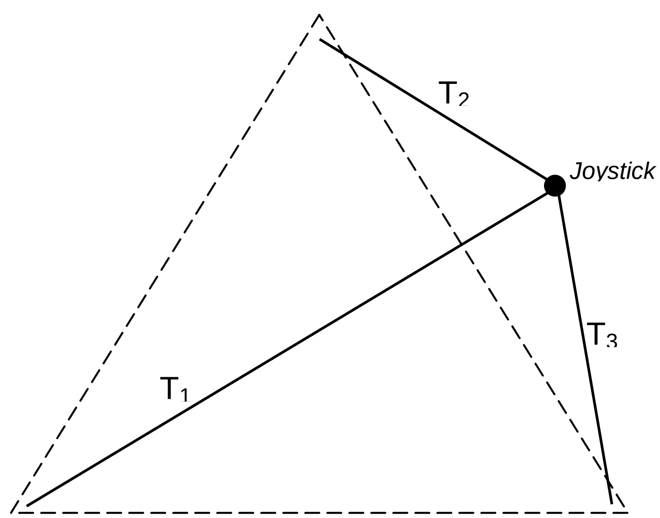


Figure 4-9: Example of a joystick position outside of the allowable work area

As shown in Figure 4-9, the motors would not be able to negotiate the joystick to this point on their own. In this particular example, the tensions in the second and third cables could probably be maintained on their own, but the first cable would have to remain straight and rigid for the compressive force that would be applied at the joystick given that the other two cables were in tension. Since a cable can not handle a compressive force, this is not an option. It is also clear that the minimum tension required on all cables (discussed in detail in the previous section) can not be maintained. Even given the possibility that there is somehow tension on two of the cables, there is no way to have tension on the third.

However, in some cases the user's applied force allow for the possibility of motion to occur outside of this triangular work area. The user acts as another actuator (operating from the joystick in Figure 4-9), and is the mechanism for maintaining tension on all cables. This new work area is limited either by the cable length or the frame that encloses the robotic mechanism. If the cable lengths permit it, it is necessary to include forbidden areas in this new larger workspace to keep the lower base from hitting either the spools or the exterior frame. Either condition could damage the mechanism during the collision, as well as causing unplanned forces to be felt from the joystick. These points must act as singularities even though they are mathematically allowed. Equations 3-1 to 3-6 allow for all movements within this work area since it is defined by the Cartesian space given in Figure 3-5, though the conditions for 3-6a and 3-6b must include absolute values.

Problems also arise when a movement around a spool or very distant from the regular work area (if not limited by a frame like in this system) is desired, as some of the examples shown in Figure 4-10. The result is cable entanglement (either with other cables or with a spool and lower base) and alterations to the cable length equations to account for longer lengths needed to compensate for the collisions. This case becomes fairly complex and is not relevant this project. These instances can never occur is because it is physically not possible given the constraint of the outer frame. While some error is normally introduced as the cable leaves various sides of the spool during movement, it is a small error and not as significant as those shown in the figure below.

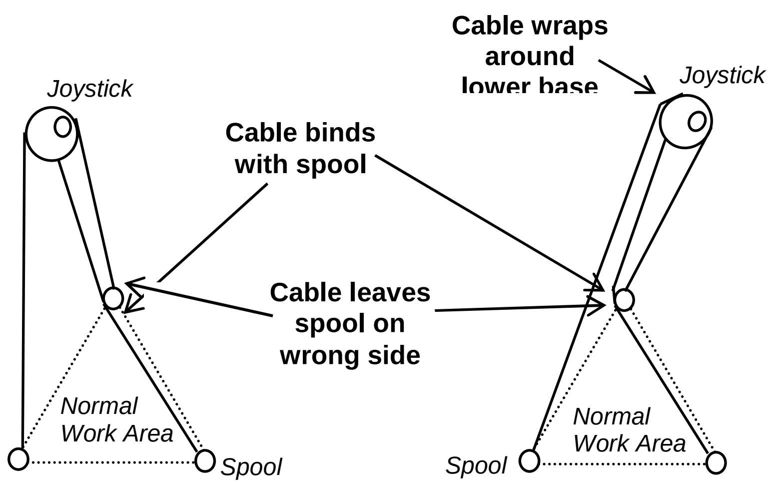


Figure 4-10: Examples of positions that would cause significant errors (no frame limitation)

During operation in the area outside of the triangle, there is a possibility of cables losing tension. Should the user move quickly from a the position shown in Figure 4-9 toward the regular (triangular) work area, there is no way to prevent or slow this motion. In the example as shown, the first cable would have to remain rigid under compression in order to hinder any movement toward the inside of the figure. To avoid a problem, the tension sensing must be quick to note the decline in tension and the motors must move quickly to regain the tension. Since this system is designed for slower movements, it is possible that the motors would fail to be sufficient in such a case.

In the case of this device, none of the areas outside of the aforementioned triangular work area are possible. This is achieved by using cables that are long enough only to reach the furthest edge of the work area. Only collisions between the spools and the lower base need to be considered when defining forbidden areas for the user's movements.

Chapter 5

SYSTEM IMPLEMENTATION

This chapter will detail the physical construction of the device. Information about the parts used and their implementation are provided, including integration of the control system. The ultimate goals of the device when selecting and implementing components into the system were kept in mind. Components were selected based both upon cost and ease of implementation and repair. The need for later additions of force sensors and haptic feedback were accommodated. Also, aesthetic considerations were kept in mind to make the system more appealing to a perspective user. For instance, all circuitry and cables were mounted and run along the underside of the desk and in the drawer to keep them out of site, but still easy to maintain. This not only made the system appear neater, but has the added feature of being safer since parts with exposed electrical current are not easily accessible. The drawer-mounted circuitry aids in maintenance by keeping all of the circuitry in one central location that can be pulled out and inspected or repaired easily.

5.1 Construction

The construction foundation is a plain writing desk. The device does not require the use of any particular type or style of table, provided that the top is positioned at a comfortable height for the user. The particular desk used in this project blends in with typical household furniture and also features a drawer that safely houses the circuitry.

Three holes, slightly large than the motor shaft, were drilled into the top of the table to allow the shafts to protrude from the top. Four smaller counter-sunk holes were drilled for each motor for the mounting bolts such that they do not protrude from the table top and obstruct motion.

A spool was created for each motor shaft. These were made from aluminum and feature a center hole that fits snuggly over the motor shaft. A single grove along the side provides a channel to guide the cable during winding and unwinding. Each spool was also furnished a set-screw to inhibit its spinning about the shaft and a mounting hole to secure the cable against slippage. The final result is shown in Figure 5-1.

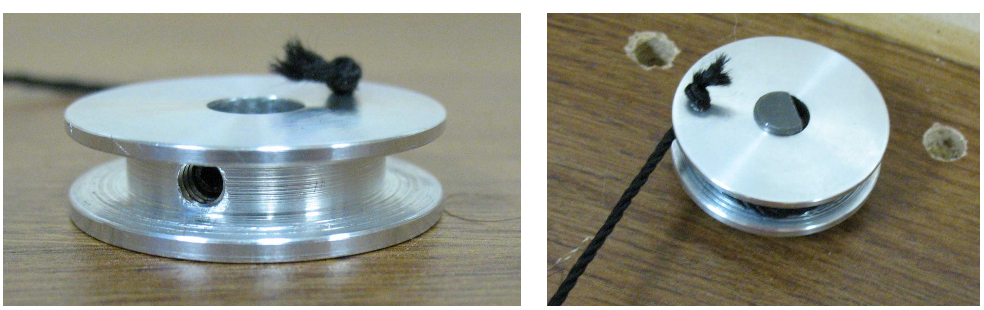


Figure 5-1: Spool for guiding the cable winding

A wooden frame was constructed around the work area to support the acrylic platform. The frame was constructed out of molding stock so that it included a ledge around the outer edge of the frame to secure the acrylic sheet above the table top, as shown in Figure 5-2. The frame was cut such that the acrylic sheet is held at a height that allows the lower base to fit snuggly underneath. This keeps the lower base from moving vertically off the table top resulting in the cables being pulled out of their grooves on the spools.

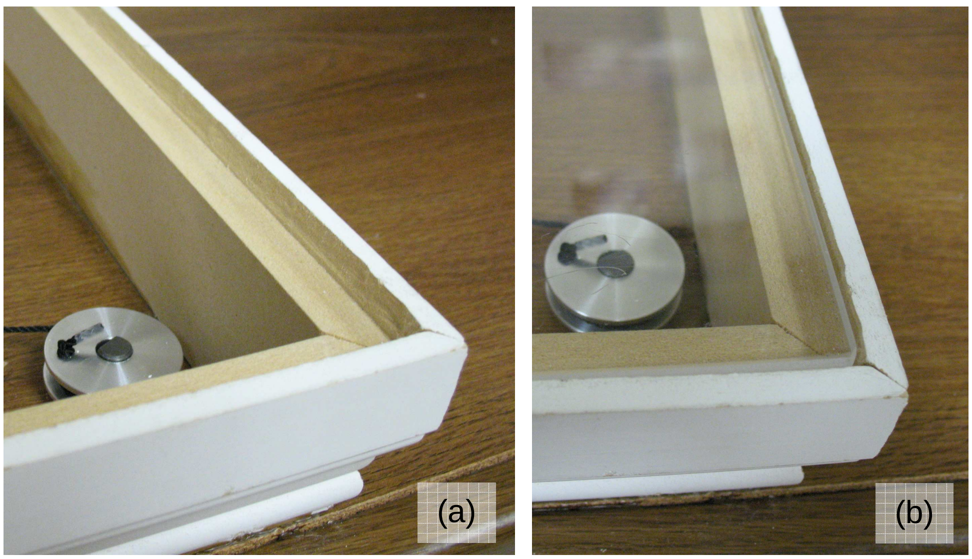


Figure 5-2: Frame for acrylic platform (a) without the acrylic sheet and (b) with the acrylic sheet

The upper and lower bases were constructed from wood. A short dowel mounted perpendicularly to the upper base became the basis for the joystick. Three rectangular recesses were carved out of both the lower and upper bases to allow the magnets to fit flush with the base surfaces. Six magnets were then glued with epoxy into the recesses (three on the upper base and three on the lower base) as shown in Figure 5-3a. The magnets had to be positioned such that the upper and lower bases were attracted to each other, rather than repelled. An epoxy was then used to fill in the remaining gaps and seal the magnets flush into the recesses. After curing, some sanding insured that the two surfaces were smooth. Pads of non-stick material were then glued over the both magnetic surfaces to allow smooth motion over the acrylic sheet (Figure 5-3b). Felt was added to the underside of the lower base to make the motion on the table smooth and quiet as well. To aid in comfort, some additional felt was wrapped around the dowel on the upper base to create a cushioned joystick, and also allowed for a contact switch to be mounted to sense the user's grasp.

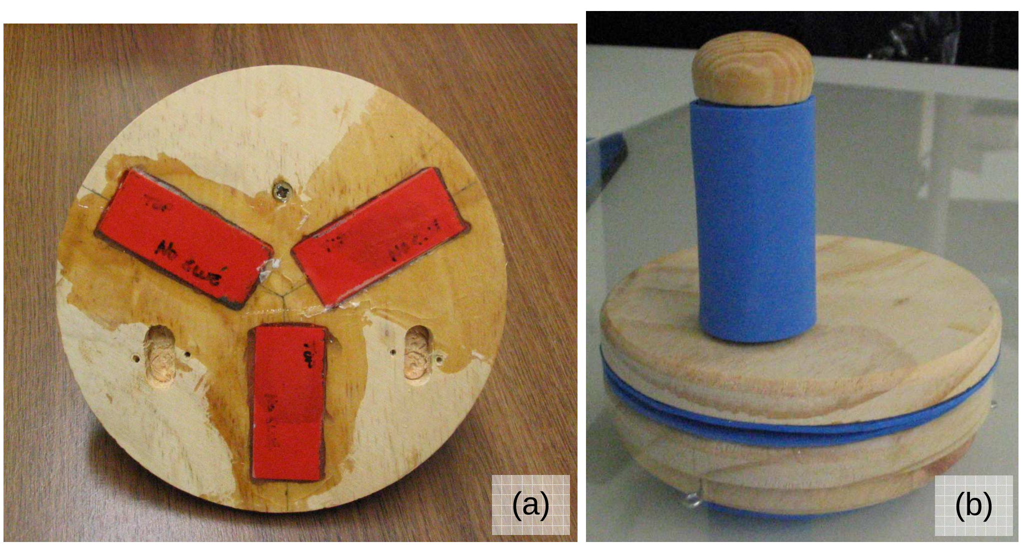


Figure 5-3: Joystick (a) magnet configuration on inside of bases and (b) assembled

Three eyehooks were inserted into the bottom platform, aligned with the height of the grooves of the spools and equally spaced. A cable, constructed from twisted nylon, was then routed from each spool to an eyehook, securing the lower base to the system, as shown in Figure 5-4. A small loop of jewelry wire was used to connect the cable to the eyehook. This was done to create an easy repair should the tension become too great since the loop will open and break, rather than the cable. The acrylic surface was placed into the frame and the upper base was placed over the lower base, bonding through the magnetic force.

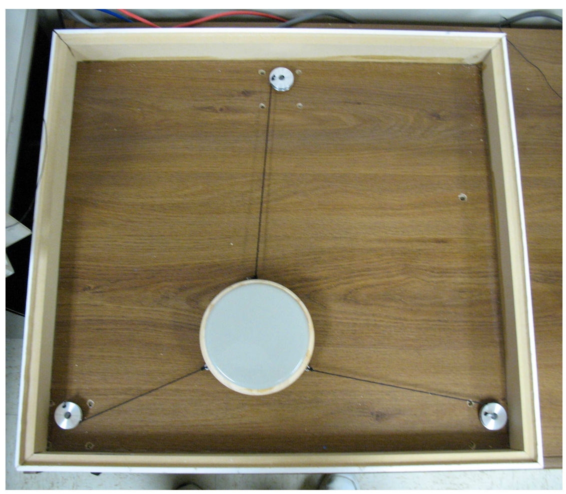


Figure 5-4: Lower base installed in work area

For diagnosis and visual feedback, a penholder was incorporated into the upper base. This allows the path to be drawn by inserting a pen and running the device while a blank sheet of paper is taped to the acrylic surface. The user can get a visual representation of their own motions by having a different color pen in place during their exercise. This completed the construction of the joystick, which is depicted in Figure 5-5.

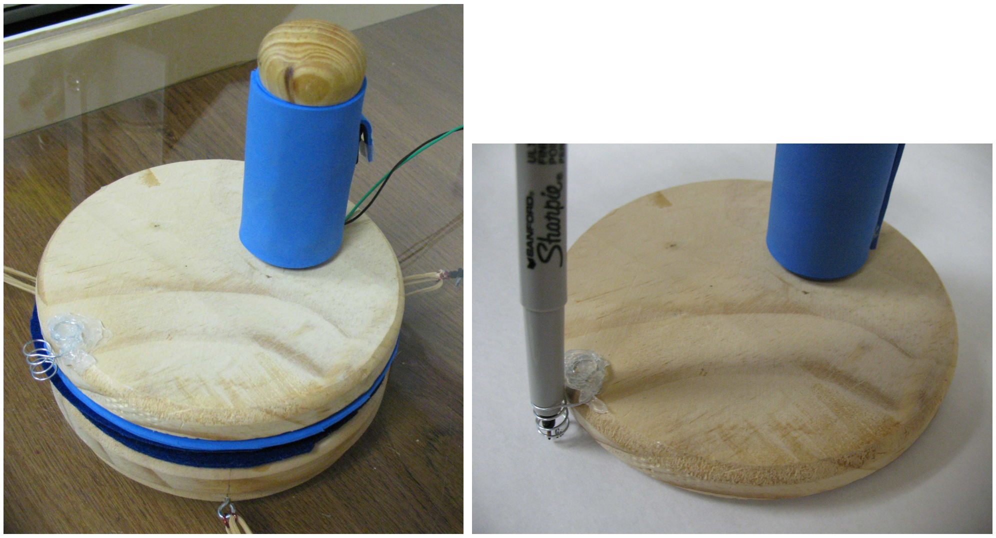


Figure 5-5: Complete joystick

Underneath the table, electrical cables from each motor were routed to supply both power and encoder signals to the motor controllers located inside the drawer. Adhesive clips aided in taking up the slack of the wires as they were routed along the outer edges of the table underside, as shown in Figure 5-6. This helps to keep the user from accidentally getting a leg entangled in these wires. The final outcome of the physical device is shown in Figure 5-7.

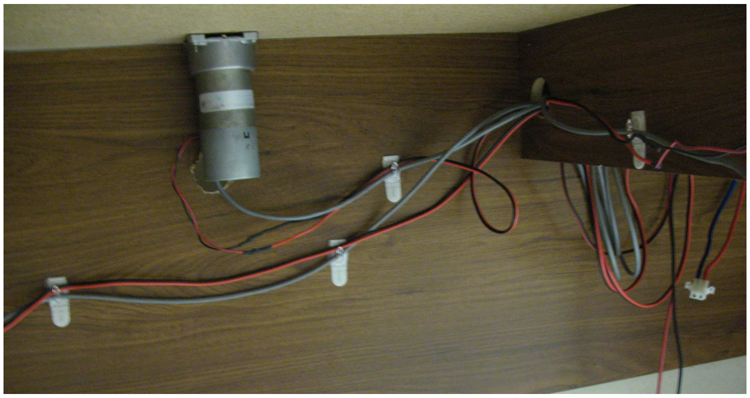


Figure 5-6: Wires routed on underside of platform

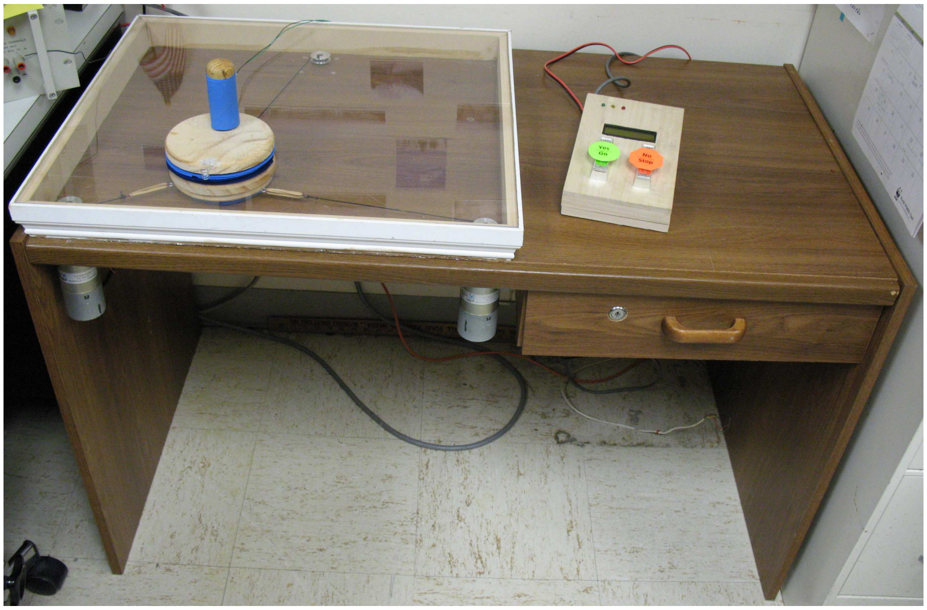


Figure 5-7: Final completed device

5.2 Electrical Circuits

Power for this device originates from a wall socket. Transformers then turn this into two different sources. The first is a DC 12-volt, 20-amphere source used to drive all three motors and the second is a DC 5-volt, lower-amperage source for driving all of the control circuitry. Both of these sources could also be derived from batteries, provided that a method for recharging the batteries was available.

The actuators used are three GMX-6MP013A geared DC motors manufactured by Matsushita Electric Ltd. Though rated at 24-VDC, they are run at 12-VDC in this system. The stall current is 4.5-ampheres. The motors include a 12.5:1 gear box and a 100-counts-per-revolution Hewlett-Packard HEDS-9100 encoder. After gearing, the actual counts per revolution is therefore 1250.

The entire system is run and supervised by a master microcontroller. This is a PICAXE-40X made by Revolution Education Ltd. This master controller handles the inputs and outputs with the user via the LCD screen and keypad. In addition, it maintains and sends information regarding the operation mode state and the current desired position. The trajectory information is maintained by three slave microcontrollers, all PICAXE-40Xs as well, which handle the individual motor operations and report on their progress. The slave microcontrollers send position data to the motor controller, an Avago HCTL-1100, which was introduced in the previous chapter in great detail. After receiving the commands, this motor controller takes care of all the calculations and encoder monitoring needed to obtain the desired position at the desired speed profile. The motor control signals are amplified through a Freescale MC33886VW H-bridge chip (formerly made by Motorola). The rotating motor shaft movements are detected via the encoders mounted and integrated inside the motor case. Each slave microcontroller polls its HCTL-1100 and monitors the status, reporting to the master controller as necessary. The emergency stop function either directly cuts power to the motors or is commanded via the master microcontroller, as detailed previously. This control architecture is shown in Figure 5-8.

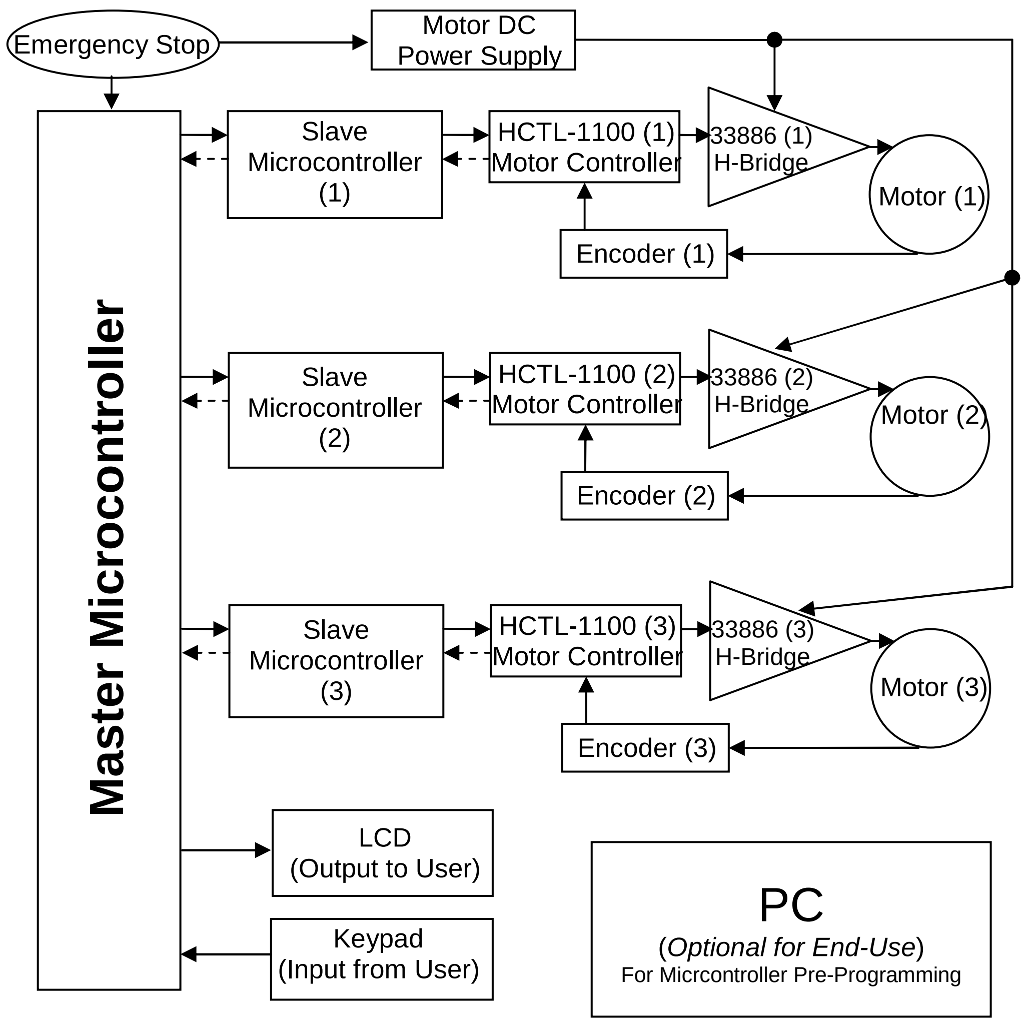


Figure 5-8: Full control architecture for guidance-only ("Play") Mode

As some of the components, particularly the motor drivers, generate a lot of heat during operation, care was taken to incorporate appropriate heat sinks and ventilation near the components. The actual components are shown, mounted inside the drawer in Figure 5-9.

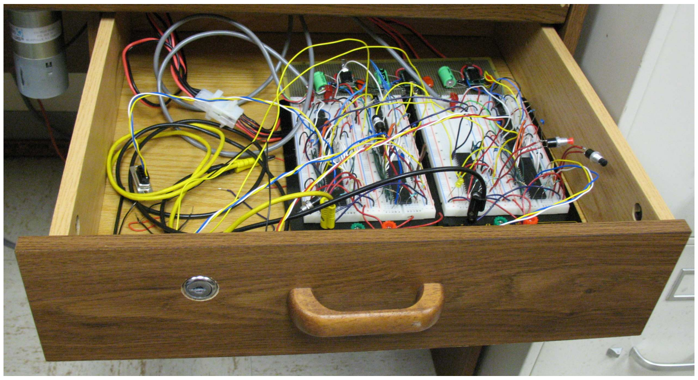


Figure 5-9: Housing of circuitry used for robot control

5.3 Software and Control

The control architecture is fairly simple in order to be fully compatible with the low-cost microcontrollers used. The basic equations and control loops were outlined in Chapter 3. All of the code used in this project can be found in the Appendix.

The master controller program was written in a version of BASIC using the PICAXE Programming Editor software, provided by Revolution Education, Ltd. This language allows for programs to be quickly produced and edited with the simple and user-friendly instruction set. Unfortunately, this also gives limited access to the internal functions of the microcontroller and thus limits the functions that the chips are able to perform. The program was written to coordinate and monitor the activity of the slaves as well as performing the interaction with the user via the LCD and keypad. Before any motion is begun, the master controller runs through the start-up procedure and makes sure that the user is ready and aware of the action that will take place.

The slave programs are generated using MATLAB, with the output programs written in the same BASIC language used for the master. The initial starting point and desired motion (given as a set of equations) are the only factors that need to be altered to generate a new path. The maximum velocity, acceleration rate, and workspace geometry can also be adjusted to personalize the program for a given user or a similar robot layout of a different size.

Creating a program to generate the slave programs required translation the geometry into the simple coordinates that the microcontrollers could utilized, along with the HCTL-1100 controllers. The first step was changing the values (given in millimeters) of the motion equations into encoder values. Using information from the encoder, motor controller equation and gearing of the electric motors used, it is known that there are 5000 encoder counts per revolution, as seen from the motor controller. Dividing the linear motion achieved by the encoder counts yields the distance traveled for every count of the encoder. As a check, the maximum lengths of the device must be considered. The lower base traveling too far would certainly damage the system as it hits and binds with a spool. If the condition is exceeded, the MATLAB code generates an error. If this condition is met, then code for intermediary positions within the given equations are generated so that the motion is achieved on the device through many small steps. The step size, like other attributes, may also be adjusted to customize the performance of the device to suit the individual user.

The master controller then works with the slaves to signal them to move to each intermediary step in unison. This is accomplished through handshaking between the four microcontrollers. The master signals all slaves to go, they then signal that they are going. If a slave fails to signal that it is moving within a reasonable amount of time, then the master signals a fault and halts the device. If all slaves move, then they give a signal when they are finished with that step. Again, the master makes sure that all finish within a reasonable amount of time. If this is not the case, then a corresponding fault is triggered and no more motions are triggered until it is corrected. Typically, no faults are found and the process of the slaves and master signaling to each other are continued for all intermediary steps until the device returns to the home position or an emergency stop is requested.

The master is continuously monitoring for an emergency stop through an interrupt pin. If this is triggered (either through a loss of grip of the joystick or the pressing of the “stop” button), then the active line to the motor controllers and amplifiers are dropped low (immediately disconnecting the power) and the slaves receive no signal to continue. The interface then prompts the user for what to do. The user could choose to continue the exercise from that point, or quit completely. Continuing only requires going through the start-up warnings, reinstating power to the motor controllers and amplifiers and signaling the slaves to continue. Quitting involves either shutting off the device (which may require realignments to start-up again) or automatically returning to home, after the user has stepped away.

Any errors or stop signals are displayed on the LCD screen. Most errors, except for a user-requested software emergency stop, require that the device be checked and corrected. It is expected that either the therapist or a service technician will perform these repairs.

If there is a power problem at the master controller, the device will automatically power down. The master chip must be properly powered and functional in order for the motor controllers and amplifiers to receive the signal to operate.

5.4 Human Interface

Since this robot is to deal very closely with a human patient, it is imperative that the device be made as user-friendly and non-threatening as possible. The use of feedback devices including sound and visual cues is necessary to aid in the interaction. The user must be able to get information from the system, as well as give instructions and desires. Warning and announcements of movement and errors are important aspects.

The LCD used is a BPI-216 made by Scott Edwards Electronics, Inc. It is a simple two-line, 16-characters-per-line black on green display. It has the option of using a backlight function, which can be turned on if the user so desires. The LCD choice was simply due to cost concerns and any display device could be used. For many older patients, a larger LCD screen or audio output would be a better alternative.

The keypad was designed to be as simple as possible. It consists of only two large buttons. It was determined that too many or too small buttons would be difficult for someone with even a mild impairment to operate. One button, clearly marked green and labeled, is used to respond affirmatively to any question or request asked via the visual display. It is also used to select a choice or to start the operation of a program sequence. A second button is used to decline an option, cancel an input or stop the device or operation. This second button is red and in a prominent part of the user interface area for ease of access.

The user interface devices are all mounted next to work platform on the top of the desk in a separate enclosure. Due to the structure of the desk, it was mounted on the right of the platform. If it were more convenient to the user, it could be mounted to the other side if a different desk or platform were used. This user interface area is built into a small box that is mounted at a slight angle so that the LCD and buttons are easier to read. This box is hinged and can be opened for easy servicing or upgrades, but should generally remain closed and inaccessible to the user. The speaker and status LEDs are also mounted here so that the user can quickly get all device information from one place.

The final item was the hard emergency stop button. This button shuts down the system in the case of an extreme error. Due to the weight and current moving through the switch, it is mounted directly into the tabletop. This ensures that the switch is offset, but still easily accessible at all times and can handle the large force of someone possibly hitting it quickly during a problem. It also ensures that the higher-current main power wires that it connects are not easily accessible to anyone.

The user interfaces are shown below in Figure 5-10.

Figure 5-10: User interface enclosure

Chapter 6

RESULTS

The completed system demonstrates the feasibility of this type of device. While it does not fully implement all possible modes of operation and leaves room for many future improvements, it does fulfill the goals of developing a practical home-based therapy system with marketing potential.

6.1 Evaluation of Performance

The robot fully functions and achieves the main goals set forth for the Play Mode. The end-effector moves through the given path as expected. The joystick, connected via magnets, tracks the lower base at slow speeds with either no or a small delay and a maximum offset during this delay of about five-millimeters. The nylon cable used has a sufficient amount of spring such that it can be used alone for slow speeds, though springs are used to accommodate the slight inaccuracies during motion sequences for large shapes and faster speeds.

The acrylic shield and magnetically-attached joystick function just as expected. It effectively transmits a reasonable amount of force (up to the break-away force) and can thus adequately move an impaired user. The force at which the joystick and base separate are adjustable through several factors including the choice of magnets, number of magnets and their arrangement, the material used between the magnets and the distance between the magnets defined by the acrylic layer and sliding pads. This allows for the device to be customizable for a wide range of patient abilities. The prototype as describes has a break-away force of approximately 1.5 kilograms in any direction.

The overall device appearance is compact and fairly aesthetic. It can easily blend into a home environment with careful consideration of material colors.

Robot Performance

To gain a quantitative result of the performance, various patterns were programmed and the robot's output was traced using the pen holder. The initial pattern programmed was a small circle, with the paths from the home position. The program was then run for different maximum velocities. The results are shown in Figure 6-1.

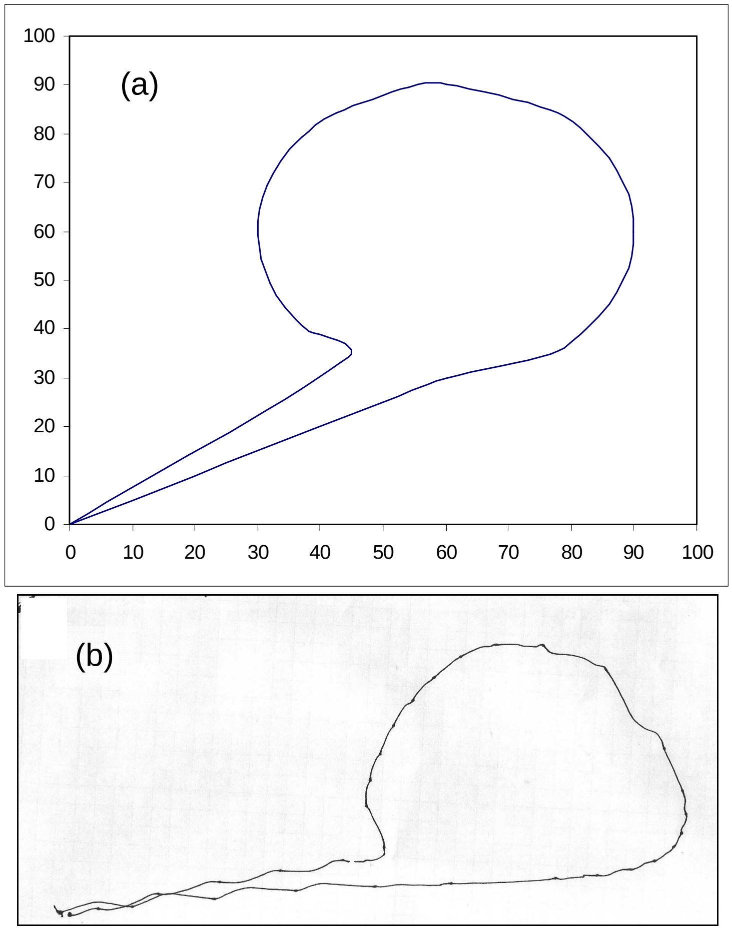


Figure 6-1: Circle pattern output
(a) Excel graph of pattern (b) Actual shape traced by robot

The figure above shows the output. The Excel graph shows the scale in millimeters. The circle size and shape are as desired, within the steps specified in the software. The circle does not fully close due to the large step size used for intermediary positions that were defined in the program. Unfortunately, a smoother pattern could not be obtained. This is due to limitations within the microcontrollers. The solution for this is the either incorporate external memory to store the intermediary point date or to upgrade the controllers. The newest versions of the microcontrollers allow for the calculations to be made for each point within the chip, which is ideal since positions can be calculated in real time during all modes of operation.

Other shapes were traced as well and are shown in the figures below.

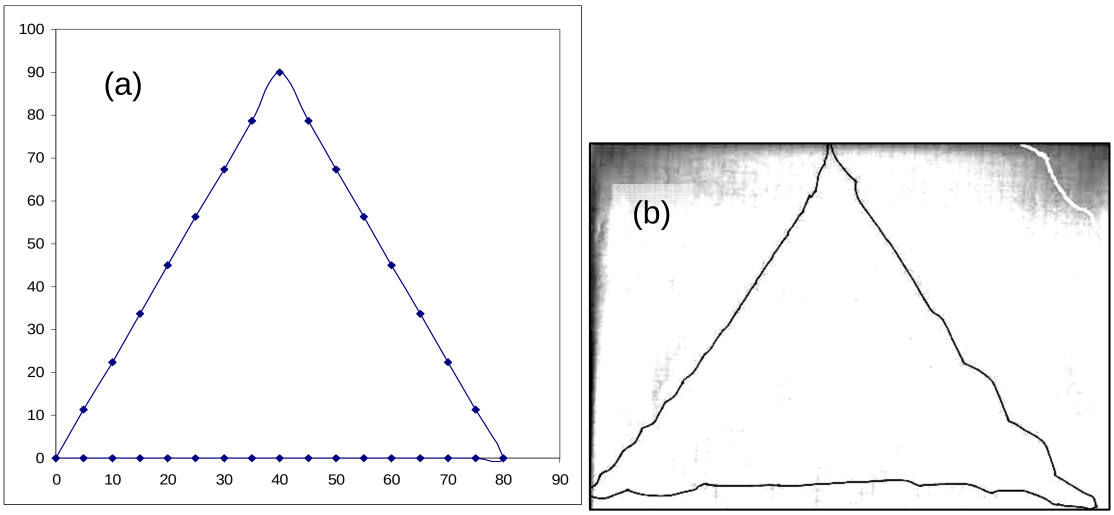


Figure 6-2: Triangle pattern output
(a) Excel graph of pattern (b) Actual shape traced by robot

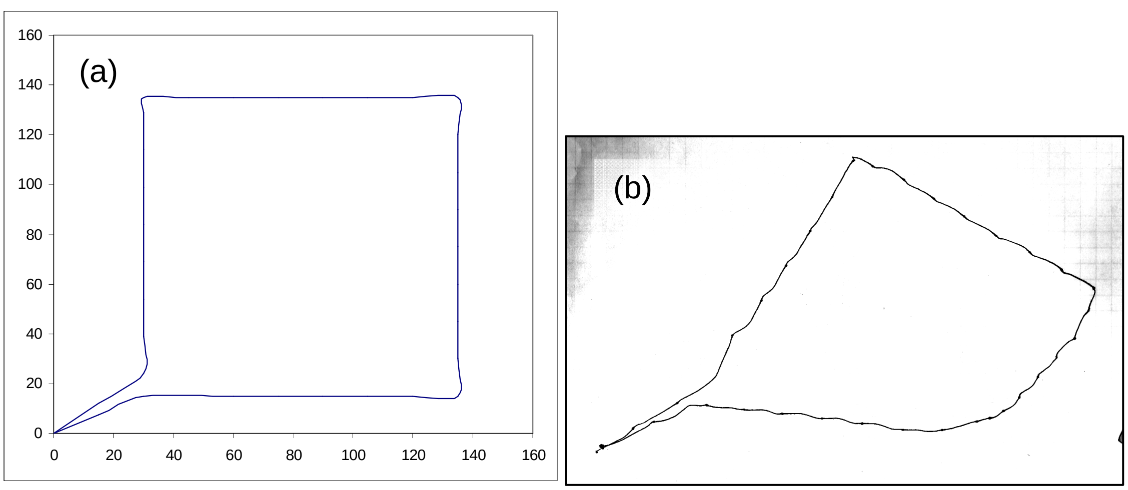


Figure 6-3: Square pattern output
(a) Excel graph of pattern (b) Actual shape traced by robot

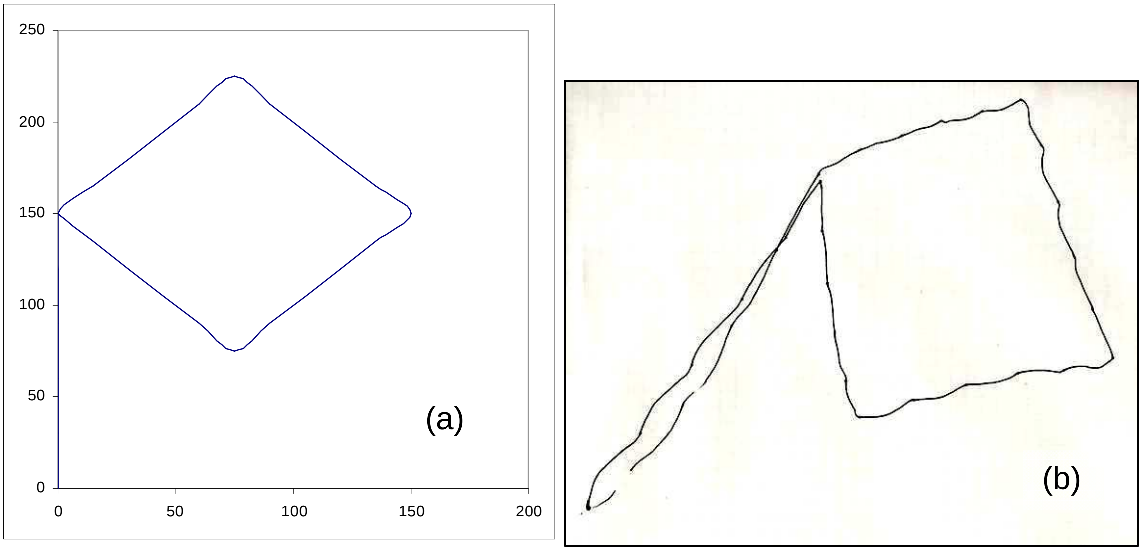


Figure 6-4: Diamond pattern output
(a) Excel graph of pattern (b) Actual shape traced by robot

As the above figures show, the robot does a fairly accurate job of tracing the patterns. There are two evident problems. The first is that nearly all patterns are slightly tilted from the programmed version. The second is that the square pattern shown in Figure 6-3 did not trace as well as those in the other figures. The cause of both problems appears to be the same.

The robot proves to be less accurate when working near the edges of the defined work area. The triangular work area forces shapes that barely fit into the work area, such as the square example in Figure 6-3 to become distorted. Once this happens, it takes some time for the cables to properly realign and find the correct balance. In the case of the square, this does not happen until the shape is being completed and so the bottom portion is completely inaccurate. The triangle shape shown in Figure 6-2 does not display any significant errors. This shape keeps a constant distance from all sides of the work area and is the best performing shape because of this fact. A diamond shape is shown in Figure 6-4. Like the triangle, the edges of the diamond shape maintain a relatively constant distance form the edge of the work area. The only major error in this shape occurs in tracing the straight line from the home position to the start of the shape. As with the square, this part of the path is not handled as accurately as it approached the edge of the work area.

Overall, shapes can be made with sufficient accuracy for physical therapy providing that the shape is well within the work area, or runs parallel to each side that it nears. Small shapes with small step sizes between the calculated intermediary points perform better than larger shapes and larger step sizes as would be expected. The size of the shapes, provided that they are not "clipped" due to being too close to the edges are within a few millimeters of the expected size. The inaccuracies and related skewing of the shapes (as in the square) are the only cause for pattern size errors.

Tests were also done to determine the robot performance at different speeds. The output of the circle at various speeds is shown in Figure 6-5.

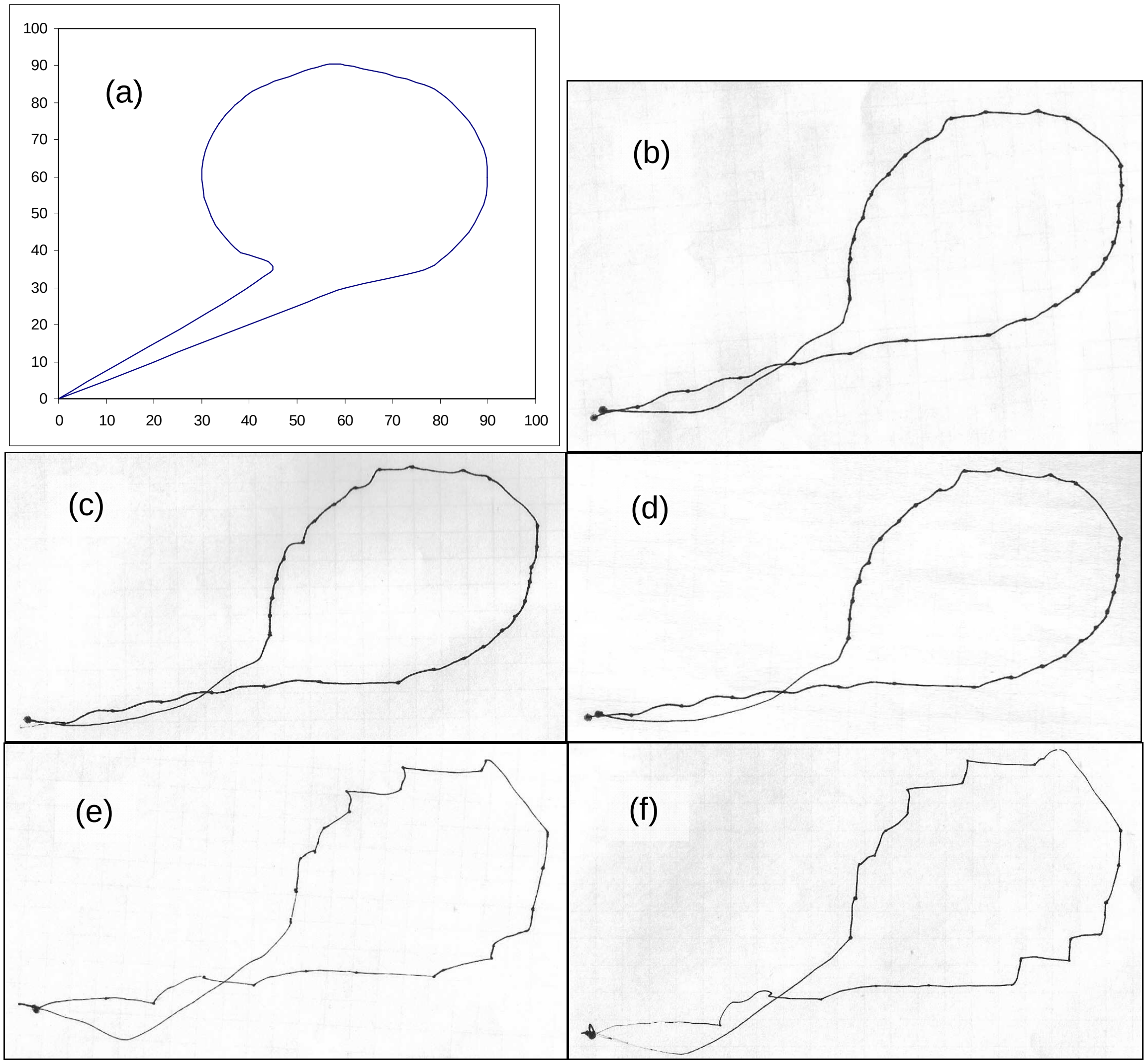


Figure 6-5: Actual robot circle pattern output for various maximum velocities (a) Excel graph of pattern; Traced output at: (b) 10% of the maximum velocity (c) 25% of the maximum velocity (d) 50% of the maximum velocity (e) 75% of the maximum velocity (f) 100% maximum velocity

The results in Figure 6-5 show that the velocity of the system impacts the accuracy of the output if it is increased above a certain threshold. For speeds below 50% of the maximum allowable system velocity, the path is followed with about the same degree of accuracy. The speed preference could therefore be adjusted by the user or therapist to suit the goals of the therapy session. More than 50% of the maximum velocity yields somewhat unstable results. These velocities cause the path to be somewhat jerky with more pronounced direction changes and errors do to overshooting the target position. This is both uncomfortable to feel as well as a poor example for muscle training. Since the intended users will have impairments, the requirement for slow system speeds is likely not to be an issue and should even be highly welcomed.

To correct the position errors, there are two possible options to explore. One is to use a different motor controller or a different control profile. The trapezoidal speed profiling does not permit a derivative term for control and so this limits the accuracy. A second option is full implementation of force sensors. For instance, adjusting the cable length to maintain constant tension may allow for the system to recover quicker when a path is not followed accurately. The system can then self-correct these errors and possibly increase the accuracy near the outer edges of the work area.

6.2 Limitations and Requirements

While the device functions sufficiently, some changes would allow for better operation. In addition, there are a few limitations and complications of this device, arising from the simplicity.

In the current condition, tension of the cables can not fully be measured. This creates several problems. First of all, three of the four modes of operation are not readily feasible. Without knowing the forces the user is exhorting, the device can not properly react or adjust. While this is not a major issue for the Play Mode, tension data could improve performance since this alters motor operation.

Without force sensing, one major problem that could arise is that one or more cables may become significantly slack or in tension. In the former case, the user will be able to move the joystick outside of the intended path and the cables will slip off the spools, causing the position accuracy to be undermined. The case of too much tension is no better, as it will cause one or more cables to break away from the lower base and no longer have any bearing on the joystick position. The master controller is unable to recognize these conditions without force sensors nor does it record an error. Only by visually inspecting the system can the problem be recognized without a sensor.

The use of force sensing was investigated. To ease in data capture, a Personal Measurement Device PMD-1208FS was used to read the data and send it to a virtual on-screen oscilloscope via a USB connection to a personal computer. In the final implementation of the device, the measurements could be sent to either a PC or the microcontrollers, depending upon the desired system speed or cost.

Current Sensing

Force measurements were first attempted through the use of the motor current. The current used by the motor increases as motion resistance increases. In this case, the motion resistance is a factor of the cable tension.

Since it is easier to measure a voltage than it is a current, a low-value shunt resistor (with high current capacity) is put in series with the motor. The voltage change across this resistor is considered in determining the current flowing through the circuit. The current can be found by dividing the voltage measured by the resistance value used. In this case, a one-ohm resistor was implemented in order to maintain a high enough voltage at the motor. It was found that there was too much high frequency noise to get an accurate reading, so a capacitor was added to create a low-pass filter, shown in Figure 6-6.

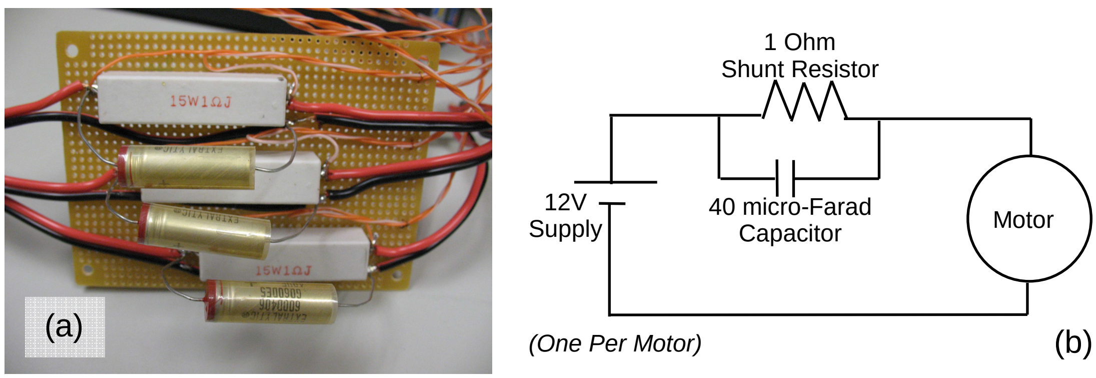


Figure 6-6: Current-measuring board with filter

As shown above in Figure 6-6b, the shunt resistor was first used in series with the motor to be used in calculating the current. To suppress some of the system noise, a capacitor was put in parallel in order to create a low-pass filter. The time-delay introduced through this filter was not significant considering the slow processing speed used to sense the change in current. It allowed plenty of time to respond. The actual filter circuit, shown in Figure 6-6a was made to be easily removed from the system, if needed.

The filter and measurement circuit gave a good reading, but the voltage difference measured was insignificant compared to the remaining noise, as shown in Figure 6-7.

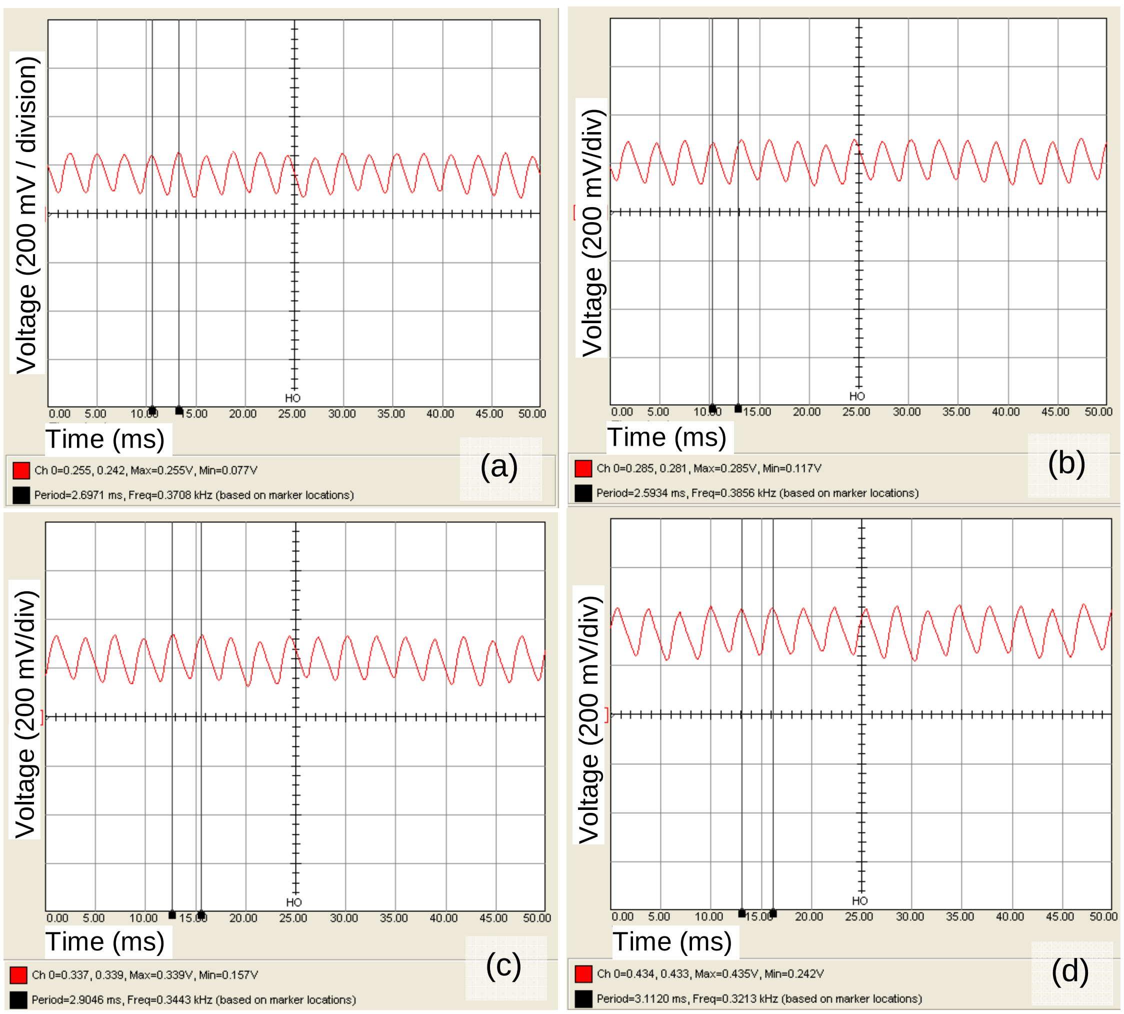


Figure 6-7: Voltage readings of shunt resistor force sensor (a) with no load (b) with a 0.5-kg load (c) with a 1-kg load (d) with a 1.5-kg load

The outputs show that the sample that a microcontroller would read at any time did not uniquely describe a force within 1.5 kg. Since a stall and separation of the upper and lower bases occur at this point, the useful range of measurements from this set-up are not within the range needed for this device. The voltage for no load is in the range of 77- to 255-mV as shown in Figure 6-7a. The range for the full load (1.5 kg) was between 242- and 435-mv. A controller reading a value of about 250 would not be able to make a clear judgment of the force acting on the motor. It was determined that this was not the best way to gain tension information.

Strain Gauges

The use of force sensors on the cable to provide tension information showed more promising results. The biggest difficulty with this is routing the wires from the sensor to the controller. Since the cables and end-effector are in constant motion, keeping the wires out of the way is a challenge. In addition, the forces on the signal wires as the sensor moved caused inaccurate measurements to be taken.

The best solution, though it requires some redesign of the current mechanism, is to put a measurement device on the table rather than in-line with the cables. This requires that the sensor be mounted between the spool and lower base such that it is in-line with the cable. This way, it is held stationary and can get a more accurate reading of the tensions. This is how it is implemented on many systems, and is shown clearly from the SEGESTA system in Figure 6-8 [17].

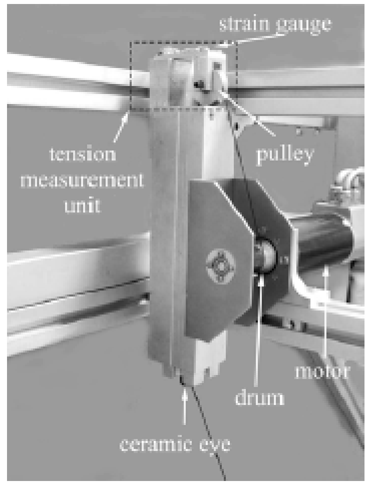


Figure 6-8: Mounting of tension sensor (strain gauge) on the SEGESTA system

An attempt at an inline strain gauge was as a test for this project. The gauge was made from a small piece of plastic bent into a “C-shape” with a pair of Micro-Measurements foil gauges mounted to both sides, as shown in Figure 6-9.

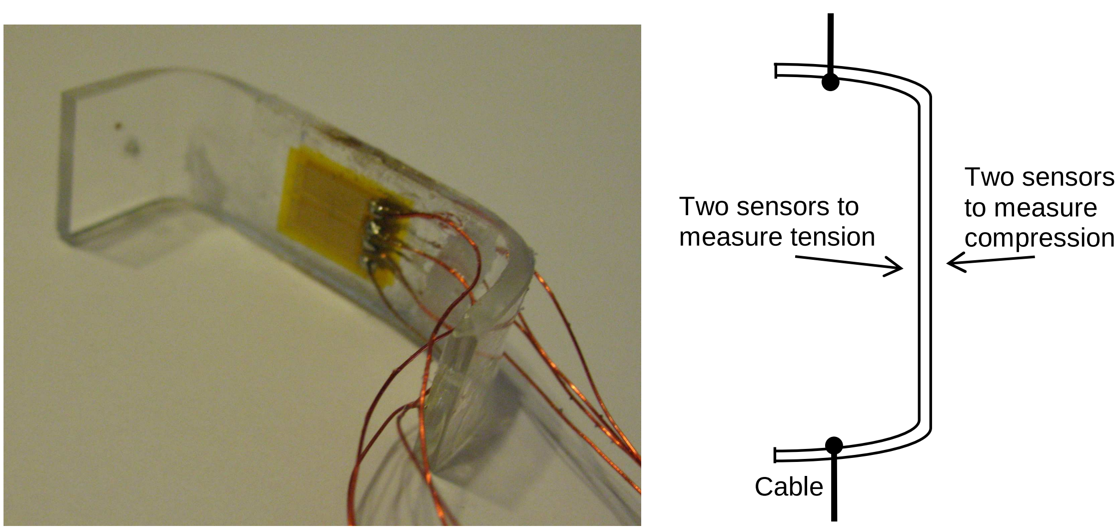


Figure 6-9: Strain gauge for mounting in-line with cable

The result is that a balanced Wheatstone Bridge can be read from the two compressed and two stretched sensors as shown in Figure 6-10. To facilitate the gauges without reconfiguring the entire robot layout, the gauges will be mounted on the wires. This does create some problems as mentioned previously, however. To limit the problems, the amplifier can be attached directly to the sensor as this limits the number of wires that must be routed off of the sensor and the mechanism from eight to four. This is shown in Figure 6-11. The overall layout of the sensors is shown in Figure 6-12.

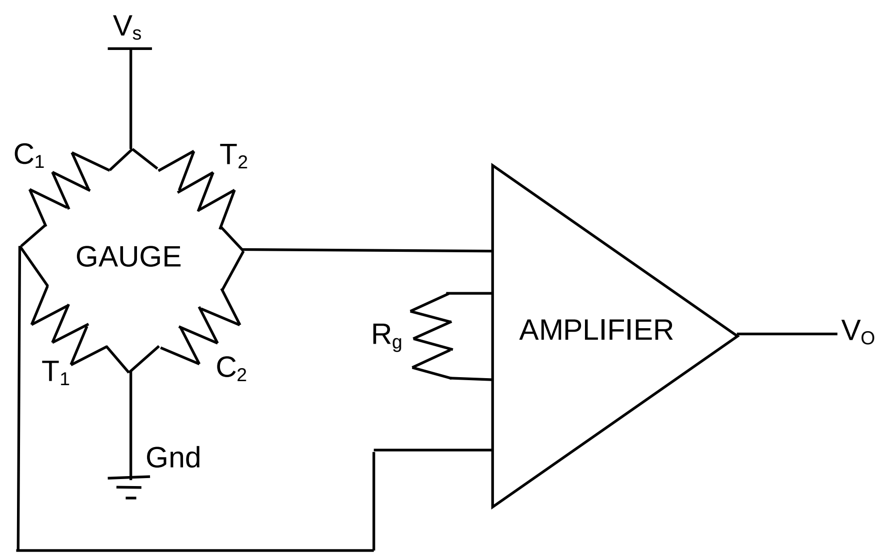


Figure 6-10: Basic strain gauge circuitry

The figure above shows the basic circuitry for the strain gauge. The four resistors in the Wheatstone Bridge on the left size are applied to the surface of the C-shaped material as shown in Figure 6-9. The resistances are labeled according the compression or tension measurements that they experience as shown in the previous figure. Resistive sensors on the same side of the strain gauge are mounted to opposing sides of the bridge in order to provide natural balance. The bridge is powered by a supply as shown. Ideally, an appropriate precision instrumentation amplifier is used to amplify the signal from the bridge, as depicted on the right. These typically have an external resistor, R_g, which is used to set the gain of the amplifier.

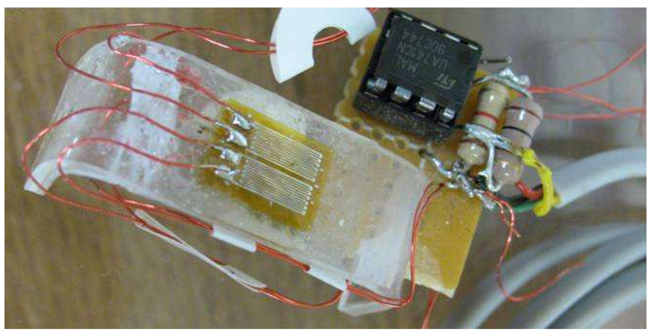


Figure 6-11: Strain gauge amplification circuitry mounted to gauge

The gauge above is an example of how the weight of the wire was reduced by adding the amplifier to the gauge. The weight of the wires pulling on the sensor during movement and the complexity of routing the signal wires through the cable mechanism to the control circuitry had been causing inaccurate readings before this modification. This allows for only three or four wires (depending upon the amplifier) to require routing from the gauge.

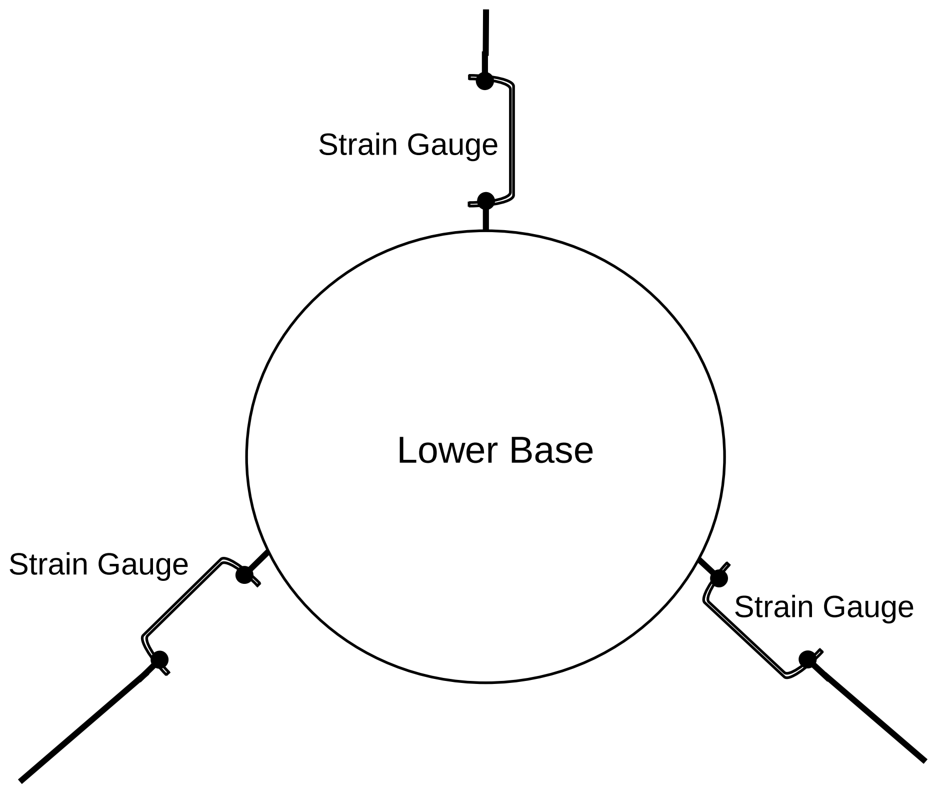


Figure 6-12: Planned set-up for strain gauges in-line with cables

One gauge was tested without amplification of the signal to determine if this type of gauge would be suitable for the project. The results are shown in Figure 6-13.

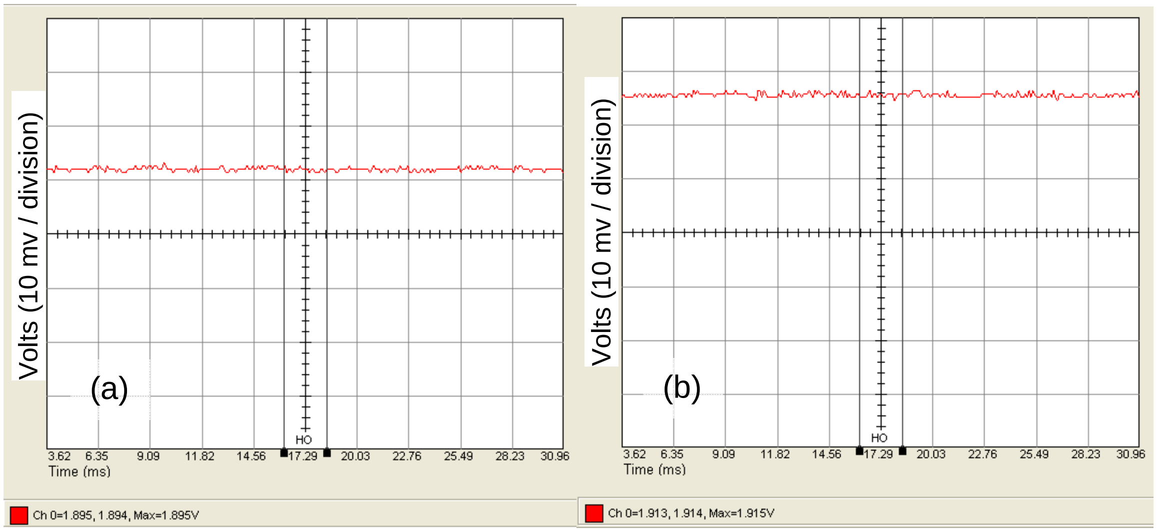


Figure 6-13: Output of strain gauge for (a) no load and (b) 0.5-kg load

As seen from Figure 6-13, there was approximately a 10-mV difference between the unamplified signal of the strain gauge with no load and with a 0.5-kg load. While this is not much, the noise was very small and the difference between the two signals is quite clear. With the use of proper amplification circuitry, the signal could certainly be used to determine the force on each cable. These gauges would not add a significant cost to the system and would allow for the full functionality desired to be implemented. Full implementation of the force sensors it out of the scope of this initial stage of the project, however.

With a strain gauge fully implemented on all cables, the master controller can monitor and signal for change should a cable become too taunt or slack. The controllers simply need to compensate, either through a reduction or increase in the motor position command, to maintain a constant tension on all cables during Play Mode. For the other modes, a measurement of the cable tensions leads to a calculation for the overall reaction of the system, whether it be moving in the same direction as the user or applying more force against the motion.

The addition of tension measurements also solves another shortcoming of this prototype. The homing of cable-driven devices that are suspended has already been solved [42]. However, this robot is not suspended and any motions without force sensors could lead to incorrect cable tensions and damage. Therefore, it is not possible to automatically home the robot during the start-up or after an error. In order to start-up or recover from many errors, including full power loss during operation, the robot must be reset to home by shutting down power and manually adjusting the motors and end-effector so that the position and tensions are correct. An impaired patient is certainly not capable of such a task and it would prove to be troublesome for relatives and therapists as well. This is because it involves removing the acrylic barrier and handling the mechanism, so an accidental powering of the motors could prove to be a hazard for an unobservant individual. The best solution is for the machine to be able to automatically reset itself to a home position and verify this upon start-up. With the addition of strain gauges, the robot could move all motors so that the cables move the lower base to the home position, while maintaining the appropriate tension. The home position is reached by the lower platform triggering a sensor mounted at the kinematic origin of the table surface.

Suggestions for Further Improvements

In order to improve system response and allow for better customization of the control, either a different motor controller should be used or modifications should be made to put the system into better alignment with the limitations of the HCTL-1100. Since speeding up the system is not an option due to safety considerations, then the system needs to have a lower frequency response. This would allow the system to move easier at slower speeds, which would be desirable. This can be done by using a higher gear ratio in the gearbox, using a motor with a lower voltage-rating, using a less accurate encoder (i.e. with fewer slots), using a slower motor (with a higher mechanical constant) or decreasing the spool size so that the cable speed is decreased.

Increasing and personalizing the interaction with the user is another important area where improvements should be made. There are many items that could be added to the device to provide user feedback. Ideally, a more “friendly” user interface would later be developed to ease any anxiety while using the robot, rather than just and LCD and keypad. Since cost is a major consideration, only the simplest, lowest-cost devices were implemented in the presented prototype.

One low-cost option to increase visual feedback is the use of LED indicators on the joystick indicating the desired direction to move. This would require that power and communications be made to the upper joystick platform either by wires or via a battery and wireless communication hardware. The advantage would be additional visual feedback right on the joystick while completing the exercise, which is where many users will prefer to look.

Another option is the use of a computer screen. This is done with nearly all other rehabilitative projects such as the MANUS. While the visual feedback can be very descriptive and varied, there is some disconnect between the physical movement of the hand and the movement of a curser on a screen. The use of a computer screen would unfortunately greatly increase the cost, size and complexity of the system. Still, the ability to turn the activities into "video-games" and other appealing activities is a benefit. Another possibility is the use of a virtual face on this computer screen to entice the user to interact.

The use of these devices and the determination of the most effective method(s) are out of the scope of this project and there are many investigations of this type underway, as mentioned in Chapter 2. There will likely be many more options that can be considered in making this device more interactive as technology continues to progress.

A final area where further work is needed is the addition of remote monitoring and programming ability. This requires that the device be accessible through some means of communication, whether it be a phone line, wireless router or other means of sending and receiving data. This could be done either directly through the master controller or via a personal computer connected to it. The interface does not need to be very intricate, as only simple data streams need to be sent out and received. The data can be interpreted and represented in a user-friendly manner after reaching the therapist's computer. A program to interpret and present the information to the therapist, whether as a table, graphic, key data or a combination also needs to be developed in order for the system to be complete. As mentioned in Chapter 2, several other projects are focusing almost entirely on this aspect of the endeavor.

Overall, this device is a proof of concept that an affordable, attractive and safe machine can be built to allow patients to gain the ability to improve their functionality from their homes or with less supervision in an office setting. Still, more work is necessary to move this device into real homes.

Chapter 7

CONCLUSION

The current development and progress of this device has been presented. The design, implementation and results of the work to date have been detailed in the preceding chapters. It is hoped that work on this and similar devices will continue and some thoughts in this regard are presented in this chapter.

7.1 Suggested Future Directions

This work has many areas where improvements can be made. This includes the full implementation of haptic feedback, improvement of the user interface, increased portability of the device and the addition of remote monitoring of the device.

As mentioned in Chapter 6, the full implementation of force sensors would add all of the desired functionality sought for the system. It would also allow for automatic error correction in most cases and more accurate positioning in all modes of operation. Haptic feedback also increases possible uses of this system. It is definitely the first and key step to start with in continuing progress.

A design for a more compact implementation of the force sensors is shown in Figure 7-1. The sensor mechanism is built into the lower base by incorporating strain gauges on levers holding the cables on the base. This has the added benefit that should an error occur and the lower base hit a spool, the sensor will pick this up since a negative value will suddenly be felt. This design is more compact than putting the sensors anywhere else and therefore allows for a greater work area to be utilized with the same equipment.

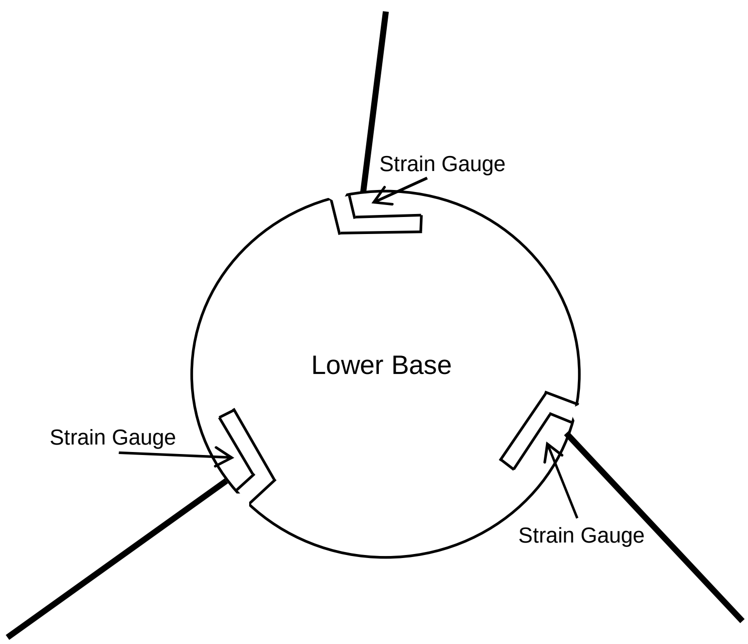


Figure 7-1: Compact sensor design

The device size is compact but allows for an even smaller design. All components fit with plenty of room to spare on the writing desk. The current device houses most of the circuitry in the desk drawer, but some components are maintained inside the user-interface box. Due to the size and placement of the user-interface box, it is possible to mount *all* of the circuitry here. This would free the drawer for other uses, or allow for a smaller simple desk to be used. Furthermore, the device can be made even smaller by enclosing the work area in a triangular frame, rather than the rectangle and using the area gained to house the circuitry and some of the control interface. The device would then be a very portable rectangular box, as shown in Figure 7-2.

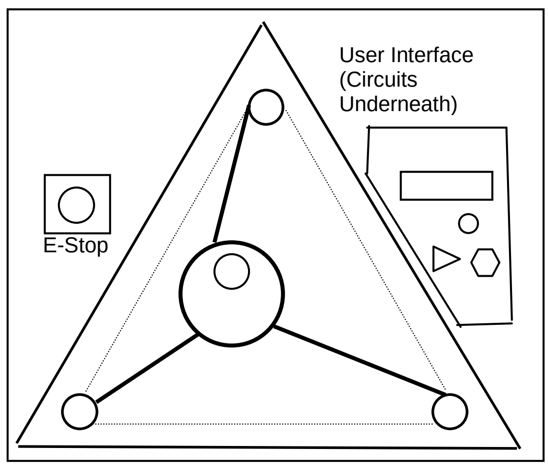


Figure 7-2: Device in a compact layout

Secondly, the use of pancake motors instead of regular DC motors is more advisable. This would avoid the protruding gearboxes on the underside of the table. In addition, it would allow for the position of the motors to be altered. Currently, the motors are mounted so that one side of the work area triangle was nearest to the user, as shown in Figure 7-3a. This configuration causes the motors to straddle the user's legs under the table. As a result, the majority of the work area is close to the user. If movements further out are desired with a larger work area, then the configuration should be inverted such that one motor is nearest to the user and the greater work area is further out, as in Figure 7-3b below. The use of a more open table or table-top platform would facilitation this rotation and adjustment at-will. The user could even position the device at any angle in between in order to customize the placement of the work area.

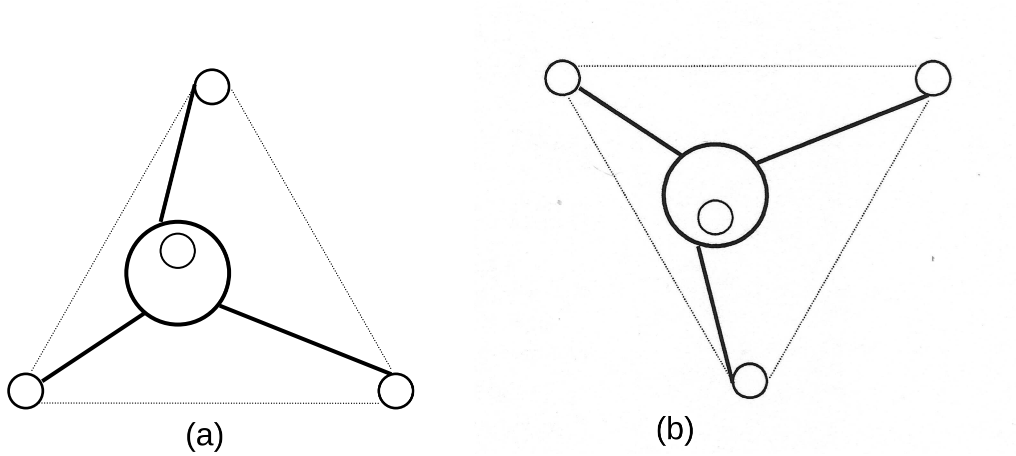


Figure 7-3: Two possible rotations of the work area

Further work in making this device portable is important for its use in any area. With the use of pancake motors, the device is not required to be mounted to a table. In fact, it could be compacted into a table-top device as depicted below in Figure 7-4. Developing this layout further expands the possibilities. The rental, storage and set-up of the device would be simplified as it could be stood on top of existing furniture and moved at will by a capable individual. The user interface area (with the LCD and keypad) could be moved to either side of the work area and secured by a Velcro, magnetic or locking connection.

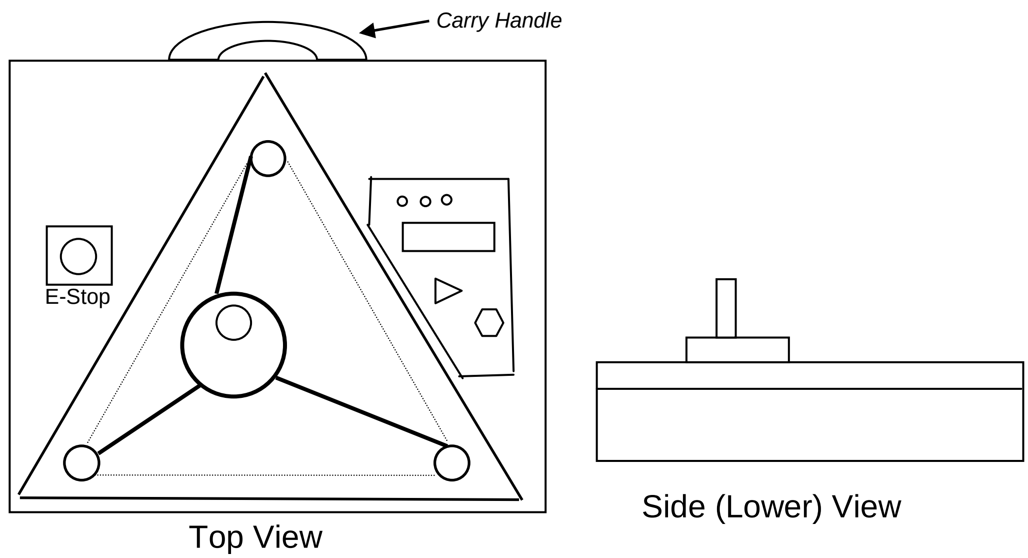


Figure 7-4: Portable rehabilitation robot

In addition, the exercise of different muscle groups or movements can be obtained simply by using a mechanism to prop the table up on an angle such that the work area is tilted, as in Figure 7-5. This opens up the possibility of movements in a partially 3-DOF motion without interaction with cables. The tilting mechanism poses some safety concerns, however, as it introduces some large pinch-points and the possibility of the device being accidentally pushed over. Ideally, it would remain statically tilted for any given exercise.

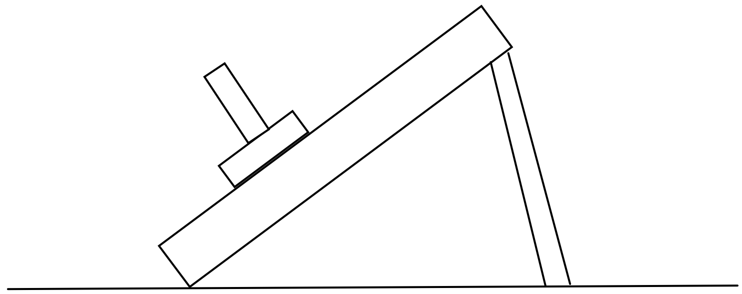


Figure 7-5: Tilted portable robot as seen from the side

The use of an electromagnet in the lower base would allow for a greater force to be exerted on the user. This would be implemented by putting a metal plate in the upper base and routing electrical power to the lower base to activate the magnet. The force of the electromagnet could then be altered at will to provide varying degrees of strength. The “break-away” force of the joystick would be easily manipulated to increase the safety, if needed or increased to allow for a greater range of haptic feedback for use in virtual reality or sports training. It also allows the magnetic force to be completely shut-down when the device is powered off or goes into an emergency stop situation. This adds another degree of safety to the device.

The use of a serial manipulator is another area where future research could head. A robotic arm that is small, light, planar and rigid could function under the acrylic sheet. As with the cables, it would be kept safely away from the user due to the barrier. Again, magnetic force would be used to connect the end-effector to an upper base with a joystick. This type of system would allow for greater accuracy (since there is almost no backlash), less error, and better routing of wires for sensors and/or an electromagnet. The kinematics and control of this type of manipulator are well-established. The cost and weight would increase, however. This rigid-mechanism type of device may share a place with a cable-driven one with different target applications. For the purpose of low-cost, low-force home rehabilitation with less emphasis on accuracy, the cable-drive still has more potential.

The exploration of further applications of this device is certainly possible. While the primary focus here was to use the device for physical therapy of disabled individuals, the platform can be modified for many other tasks. With an increase in size, strength and speed of the components used, this device has possible uses in sports training and virtual reality. Sports training would allow the device to be used much as it was presented here with modes for muscle movement training, light programmed resistance, performance analysis and motion recording. Just as in physical therapy, this would provide a precise and quantitative measurement of progress. The largest foreseeable problem is ensuring that the magnetic connection of the joystick is strong enough to accommodate an athletic individual.

Another major area of further research is the use in virtual reality systems. This could range from a simple video game to teleoperation of a remote device. Many opportunities for haptic devices in these areas are presented in available literature. The ability of a safe haptic feedback makes this a very promising and attractive device for these areas.

Finally, miniaturization and the use of more accurate sensors could allow for this device to be used to regain writing ability. In addition, it could be used as a device to filter handwriting and make it more smooth, a welcomed feature for those who suffer from tremors caused by Multiple Sclerosis or Parkinson’s disease.

Undoubtedly, new ideas and improvements will be generated and implemented. It is hoped that this system will help to spark further innovations and aid in the care of individuals in the future.

BIBLIOGRAPHY

[1] Bardorfer, A., Munih, M., Zupan, A. and Primožič, A. (2001) "Upper limb motion analysis using haptic interface," *IEEE/ASME Transactions on Mechatronics*, 6(3), 253-260.

[2] Beer, R.D., Chiel, H.J., Quinn, R.D. and Ritzmann, R.E. (1998) "Biorobotic approaches to the study of motor systems," *Current Opinion in Neurobiology*, 8, 777-782.

[3] Bosscher, P.M. (2004) "Disturbance robustness measurements and wrench-feasible workspace generation techniques for cable-driven robots," Doctoral dissertation, Georgia Institute of Technology.

[4] Bostelman, R., Jacoff, A., Proctor, F., Kramer, T. and Wavering, A. (2000) "Cable-based reconfigurable machines for large scale manufacturing," *International Conference on New Technological Innovation for the 21st Centery*, Ann Arbor, Michigan.

[5] Britton, M. and Andersson, A. (2000) "Home rehabilitation after stroke: Reviewing the scientific evidence on effects and costs," *International Journal of Technology Assessment in Health Care*, 18(3), 842-848.

[6] Bruckmann, T., Pott, A. And Hiller, M. (2006) "Calculating force distributions for redundantly actuated tendon-based stewart platforms," *Advances in Robot Kinematics*, 403-412.

[7] Bullies, K. (2005, September) "Robot rehab," *Technology Review*, 108(9), 29-30.

[8] Bühler, C. (1998) "Robotics for rehabilitation – a European (?) perspective," *Robotica*, 16(5), 487-490.

[9] Burgar, C.G., Lum, P.S., Shor, P.C. and Van der Loos, H.F.M. (2000) "Development of robots for rehabilitation therapy: The Palo Alto VA/Stanford experience," *Journal of Rehabilitation Research and Development*, 37(5), 663-673.

[10] Childress, D.S. (2002) "Development of rehabilitation engineering over the years: As I see it," *Journal of Rehabilitation Research and Development*, 39(6), (supplement) 1-10.

[11] Culmer, P., Jackson, A., Levesley, M.C., Savage, J., Richardson, R., Cozens, J.A. and Bhakta, B.B. (2005) "An admittance control scheme for a robotic upper-limb stroke rehabilitation system," *Proceedings of the 27th International Conference of the IEEE Engineering in Medicine and Biology Society*, Shanghai, China.

[12] Dijkers, M.P., deBear, P.C., Erlandson, R.F., Kristy, K. and Geer, D.M. (1991) "Patient and staff acceptance of robotic technology in occupational therapy: A pilot study," *Journal of Rehabilitation Research and Development*, 28(2), 33-44.

[13] Ebert-Uphoff, I. and Voglewede, P.A. (2004) "On the connections between cable-driven robots, parallel manipulators and grasping," *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA, 4521-4526.

[14] Edgerton, V.R. and Roy, R.R. (2002) "Paralysis recovery in humans and model systems," *Current Opinion in Neurobiology*, 12, 658-667.

[15] Emken, J.L., Wynne, J.H., Harkema, S.J. and Reinkensmeyer, D.J. (2006) "A robotic device for manipulating human stepping," *IEEE Transactions on Robotics*, 22(1), 185-189.

[16] Erlandson, R.F. (1995) "Applications of robotic/mechatronic systems in special education, rehabilitation therapy, and vocational training: A paradigm shift," *IEEE Transactions on Rehabilitation Engineering*, 3(1), 22-34.

[17] Fang, S., Franitza, D., Torlo, M., Bekes, F. And Hiller, M. (2004) "Motion control of a tendon-based parallel manipulator using optimal tension distribution," *IEEE/ASME Transactions on Mechantronics*, 9(3), 561-568.

[18] Fattah, A. and Agrawal, S.K. (2005) "On the design of cable-suspended planar parallel robots," *Transactions of the ASME Journal of Mechanical Design*, 127, 1021-1028.

[19] Feygin, D., Keehner, M. and Tendick, F. (2002) "Haptic guidance: Experimental evaluation of a haptic training method for a perceptual motor

skill," *Proceedings of the 10th Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, 0-7695-1489-8/02.

[20] Gallina, P., Rossi, A. and Williams, R.L., II (2001) "Planar cable-direct-driven robots, part II: Dynamics and Control," *Proceedings of the 2001 ASME Design Technical Conferences*, Pittsburgh, PA, DETC2001/DAC-21146.

[21] Harwin, W.S. (1999) "Robots with a gentle touch: Advances in assistive robotics and prosthetics," *Technology and Health Care*, 7, 411-417.

[22] Hiller, M., Fang, S., Mielczarek, S., Verhoeven, R. And Franitza D. (2004) "Design, analysis and realization of tendon-based parallel manipulators," *Mechanism and Machine Theory*, 40, 429-445.

[23] Hogan, N. (1985) "Impedance control: An approach to manipulation: Part I – Theory," *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 107, 1-7.

[24] Hogan, N. (1985) "Impedance control: An approach to manipulation: Part II – Implementation," *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 107, 8-16.

[25] Hogan, N. (1985) "Impedance control: An approach to manipulation: Part III – Applications," *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 107, 17-24.

[26] Johnson, M., Feng, X., Johnson, L. and Winters, J. (2007) "Potential of a suite of robot/computer-assisted motivating systems for personalized, home-based, stroke rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, 4(6).

[27] Khalili, D. and Zomlefer, M. (1988) "An intelligent robotic system for rehabilitation of joints and estimation of body segment parameters," *IEEE Transactions on Biomedical Engineering*, 35(2), 138-146.

[28] Krebs, H.I., Brashers-Krug, T., Rauch, S.L., Savage, C.R., Hogan, N., Rubin, R.H., Fischman, A.J. and Alpert, N.M. (1998) "Robot-aided functional imaging: Application to a motor learning study," *Human Brain Mapping*, 6, 59-72.

[29] Krebs, H.I., Ferraro, M., Buerger, S.P., Newbery, M.J., Makiyama, A., Sandman, M., Lynch, D., Volpe, B.T. and Hogan, N. (2004) "Rehabilitation robotics: Pilot trial of a spatial extension for MIT-Manus," *Journal of NeuroEngineering and Rehabilitation* [Available at http://www.jneuroengrehab.com/content/1/1/5].

[30] Krebs, H.I., Hogan, N., Aisen, M.L. and Volpe, B.T. (1998) "Robot-aided neurorehabilitation," *IEEE Transactions on Rehabilitation Engineering*, 6(1), 75-87.

[31] Krebs, H.I., Volpe, B.T., Aisen, M.L., Hening, W., Adamovich, S., Poizner, H., Subrahmanyan, K., and Hogan, N. (2003) "Robotic applications in neuromotor rehabilitation," *Robotica*, 21(1), 3-11.

[32] Krebs, H.I., Volpe, B.T., Aisen, M.L. and Hogan, N. (2000) "Increasing productivity and quality of care: Robot-aided neuro-rehabilitation," *Journal of Rehabilitation Research and Development*, 37(6), 639-652.

[33] Kuttuva, M., Boian, R., Merians, A., Burdea, G., Bouzit, M., Lewis, J. and Fensterheim, D. (2005) "The Rutgers Arm: an upper-extremity rehabilitation system in virtual reality," *Fourth International Workshop on Virtual Rehabilitation*, Catalina Island, CA.

[34] Mann, R.W. (2002) "Engineering design education and rehabilitation engineering," *Journal of Rehabilitation Research and Development*, 39(6), (supplement) 23-38.

[35] Mayhew, D., Bachrach, B., Rymer W.Z. and Beer, R.F. (2005) "Development of the MACARM – a novel cable robot for upper limb neurorehabilitation," *Proceedings of the 2005 IEEE 9th International Conference on Rehabilitation Robotics*, Chicago, IL, ThB01-01.

[36] Merlett, J.P. (2006) *Parallel Robots*, Dordrecht: Springer.

[37] Mihelj, M. (2005) "Human arm kinematics for robot based rehabilitation," *Robotica*, 24(3), 377-383.

[38] Ning, K.J, Zhao, M.Y. and Liu, J. (2006) "A new wire-driven three-degree-of-freedom parallel manipulator," *Transactions of the ASME Journal of Manufacturing Science and Engineering*, 128, 816-819.

[39] Noritsugu, T., Tanaka, T. and Yamanaka, T. (1996) "Application of rubber artificial muscle manipulator as a rehabilitation robot," *IEEE International Workshop on Robot and Human Communication*, 112-117.

[40] Ottaviano, E., Ceccarelli, M., Paone, A. And Carbone, G. (2005) "A low-cost easy operation 4-cable driven parallel manipulator," *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 4008-4013.

[41] Preising, B., Hsia, T.C. and Mittelstadt, B. (1991) "A literature review" Robots in medicine," *IEEE Engineering in Medicine and Biology*, June, 13-22.

[42] Proctor, F. and Shackleford W. (2002) “Embedded real-time Linux for cable robot control,” *Proceedings of 2002 ASME Design Engineering Technical Conferences and the Computers and Information in Engineering Conference*, Montreal, Canada, DETC2002/EUC-34506.

[43] Reinkensmeyer, D.J., Emken, J.L. and Cramer, S.C. (2004) “Robotics, motor learning, and neurologic recovery,” *Annual Review of Biomedical Engineering*, 6, 497-525.

[44] Reinkensmeyer, D.J., Kahn, L.E., Averbuch, M., McKenna-Cole, A., Schmit, B.D. and Rymer, W.Z. (2000) “Understanding and treating arm movement impairment after chronic brain injury: Progress with the ARM guide,” *Journal of Rehabilitation Research and Development*, 37(6), 653-662.

[45] Reinkensmeyer, D.J., Pang, C.T., Nessler, J.A. and Painter, C.C. (2002) “Web-based telerehabilitaiton for upper extremity after stroke,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 10(2), 102-108.

[46] Reinkensmeyer, D.J., Schmit, B.D. and Rymer W.Z. (1999) “Mechatronic assessment of arm impairment after chronic brain injury,” *Technology and Health Care*, 7, 431-435.

[47] Richardson, R., Brown, M., Bhakta, B. and Levesley M. (2005) “Impedance control for a pneumatic robot-based around pole-placement, joint space controllers,” *Control Engineering Practice*, 13, 291-303.

[48] Salter, T., Dautenhahn, K. and Boekhorst, R. (2006) “Learning about natural human-robot interaction styles,” *Robotics and Autonomous Systems*, 54, 127-134.

[49] Sanchez, R., Reinkensmeyer, D., Shah, P., Liu, J., Rao, S., Smith, R., Cramer, S., Rahman, T. and Bobrow, J. (2004) “Monitoring functional arm movement for home-based therapy after stroke,” *Proceedings of the 26th Annual International Conference of the IEEE EMBS*, San Francisco, CA, 4787-4790.

[50] Schall, S. and Schweighofer, N. (2005) “Computational motor control in humans and robots,” *Current Opinion in Neurobiology*, 15, 675-682.

[51] So-Ryeok, O., Mankala, K., Agrawal, S.K. and Albus, J.S. (2005) *Transactions of the ASME Journal of Mechanical Design*, 127, 612-620.

[52] Sulzer, J.S., Peshkin, M.A. and Patton J.L. (2005) “MARIONET: An exotendon-driven rotary series elastic actuator for exerting joint torque,” *Proceedings of the 2005 IEEE 9th International Conference on Rehabilitation Robotics*, Chicago, IL, WeP01-14.

[53] Tsai, L-W. (1999) *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*, New York: John Wiley & Sons.

[54] Wagner, J.J., Van der Loos, H.F.M. and Leifer, L.J. (2000) "Construction of social relationships between user and robot," *Robotics and Autonomous Systems*, 31, 185-191.

[55] Williams, R.L., II (1998) "Cable-suspended haptic interface," *International Journal of Virtual Reality*, 3(3), 13-21.

[56] Williams, R.L., II and Gallina, P. (2002) "Planar cable-direct-driven robots: Design for wrench exertion," *Journal of Intelligent and Robotic Systems*, 35, 203-219.

[57] Williams, R.L., II and Gallina, P. (2003) "Translational planar cable-direct-driven robots," *Journal of Intelligent and Robotic Systems*, 37, 69-96.

[58] Williams, R.L.,II, Gallina, P. and Vadia, J. (2003) "Planar translational cable-direct-driven robots," *Journal of Robotic Systems*, 20(3), 107-120.

APPENDIX A

Parts Specifications

Key Robot Dimensions:

Name or Symbol	Value (in mm)
m (Work area width)	274.4
n (Work area height)	237.8
D_S (Spool outside diameter)	31.25
D_{IS} (Spool inside diameter)	21.70
D_{cable} (Cable diameter or thickness)	1.0
D_{LB} (Lower base diameter)	135

Major Components:

Motor	
Matsushita Electric Ltd.	GMX-6MP013A

Motor Controller	
Avago (formerly HP)	HCTL-1100

Motor Amplifier	
Freescale (formerly Motorola)	MC33886VW

Microcontroller	
Revolution Education Ltd.	PICAXE-40X

LCD	
Scott Edwards Electronics, Inc.	BPI-216

APPENDIX B

Program Code

MATLAB Source Code:

```
% FileGenerator_Thesis2007.m by Melissa Morris
% This program generates three files for PicAxe 40X chips to control three
%  motors for the cable-driven rehabilitation device.
%   (This is the main program)

% This file creates a triangle pattern

% IMPORTANT NOTE: The initial position of the device must be manually set
%  to match the conditions put forth here!!!

% ENTER: Adjust the equation(s) for the desired path, adjust the increment
%  between intermediary steps along the path, and any adjustment for
%  physical dimensions of the device

% OUTPUT: Three .bas files are outputted that can control the three motors
%  to operate the device along the desire path

% Required Files:
%       MM2Counts_Thesis2007.m (Converts length from mm to encoder counts)
%       CreateBASFile_Thesis2007.m (Creates header for slave files)
%       AddBegin2BASFile_Thesis2007.m (Adds initial position to files)
%       AddCmd2BASFile_Thesis2007.m (Adds all positions to slave files)
%       FinishBASFile_Thesis2007.m (Adds functions, completes to slave files)
%***********************************************************************

%*** Variables ********

%File Names
Slave1File = 'Slave1HCTLControl.bas';
Slave2File = 'Slave2HCTLControl.bas';
Slave3File = 'Slave3HCTLControl.bas';
PosTableFile = 'Positions_Thesis2007.txt';

%Initial Position
initx = 0;  % mm, x-position
 inity = 0; % mm, y-position
```

```
%Platform
m = 274.4;  % mm, Horizontal length of platform
n = 237.8;  % mm, Vertical length of platform

%Kinematics
 % Desired position: (x,y)
 % Current position: (a,b)
a = 0;     % mm, x-position
 b = 0;     % mm, y-position
StepSize = 15;   % mm, space between intermediary steps

%For Program
InterSteps = 0;     % Number of intermediary steps to reach

%********************

%*** Program *********

%Create and open .bas files for each motor
PositionTable = fopen(PosTableFile, 'w');    %Create and overwrite if already there

   %Corners: 0,0 : 200,0 : 100,224
Corner1X = 0;
Corner1Y = 0;
Corner2X = 225;
Corner2Y = 0;
Corner3X = 113;
Corner3Y = 255;

%Increment steps between first two points
Slope1=(Corner2Y-Corner1Y)/(Corner2X-Corner1X);
YIntercept1=Corner1Y-(Corner1X*Slope1);
   for x = Corner1X:StepSize:Corner2X
      y = (x*Slope1)+YIntercept1;
      if imag(y) == 0
         InterSteps = InterSteps + 1;
         fprintf(PositionTable, '%f %f\n',x,y);
      end
   end

%Increment steps between second two points
Slope2=(Corner3Y-Corner2Y)/(Corner3X-Corner2X);
YIntercept2=Corner2Y-(Corner2X*Slope2);
   for x = (Corner2X-StepSize):-StepSize:Corner3X
      y = (x*Slope2)+YIntercept2;
      if imag(y) == 0
         fprintf(PositionTable, '%f %f\n',x,y);
         InterSteps = InterSteps + 1;
      end
   end

%Increment steps between first two points
Slope3=(Corner3Y-Corner1Y)/(Corner3X-Corner1X);
YIntercept3=Corner3Y-(Corner3X*Slope3);
   for x = (Corner3X-StepSize):-StepSize:Corner1X
      y = (x*Slope3)+YIntercept1;
```

```
        if imag(y) == 0
            fprintf(PositionTable, '%f %f\n',x,y);
            InterSteps = InterSteps + 1;
        end
    end

%Increment Steps Back to Home
fprintf(PositionTable, '0.00 0.00\n');
InterSteps = InterSteps + 1;

fclose(PositionTable);

% ****** Begin Creating Files ****************************

%Store the positions in a matrix
PositionTable = fopen(PosTableFile, 'r');
XPositions = textscan(PositionTable, '%f %*f');
XPos = cell2mat(XPositions);
fclose(PositionTable);
PositionTable = fopen(PosTableFile, 'r');
YPositions = textscan(PositionTable, '%*f %f');
YPos = cell2mat(YPositions);
fclose(PositionTable);

%Create files (clear and overwrite any data if file already exists)
%   and write headers
CreateBASFile_Thesis2007(Slave1File);
CreateBASFile_Thesis2007(Slave2File);
CreateBASFile_Thesis2007(Slave3File);

%Get and record current position
    %***NOTE: this is where the device should be manually set to!!!! ****
%Convert position into cable lengths
l1 = sqrt(inity^2+initx^2);
l2 = sqrt(inity^2+(m-initx)^2);
l3 = sqrt((n-inity)^2+((m/2)-initx)^2);
%Convert to encoder counts
l1encoder = MM2Counts_Thesis2007(l1);
l2encoder = MM2Counts_Thesis2007(l2);
l3encoder = MM2Counts_Thesis2007(l3);
%Append to files
AddBegin2BASFile_Thesis2007(Slave1File,l1encoder)
AddBegin2BASFile_Thesis2007(Slave2File,l2encoder)
AddBegin2BASFile_Thesis2007(Slave3File,l3encoder)

%For each position...
for i = 1:InterSteps
    CurrentStep = i;
    a = XPos(i);
    b = YPos(i);
    %Set current positions
    l1 = sqrt(b^2+a^2);
    l2 = sqrt(b^2+(m-a)^2);
    l3 = sqrt((n-b)^2+((m/2)-a)^2);
    %Convert to encoder counts
    l1encoder = MM2Counts_Thesis2007(l1);
    l2encoder = MM2Counts_Thesis2007(l2);
```

```
    l3encoder = MM2Counts_Thesis2007(l3);
    %Append new command to files
    AddCmd2BASFile_Thesis2007(Slave1File,l1encoder,CurrentStep);
    AddCmd2BASFile_Thesis2007(Slave2File,l2encoder,CurrentStep);
    AddCmd2BASFile_Thesis2007(Slave3File,l3encoder,CurrentStep);
end

%Append final information (read and write functions) to the files
FinishBASFile_Thesis2007(Slave1File);
FinishBASFile_Thesis2007(Slave2File);
FinishBASFile_Thesis2007(Slave3File);

%******* END OF PROGRAM **********************************************
%**********************************************************************
```

```
function EncoderCounts = MM2Counts_Thesis2007(MMPosition)
% MM2Counts_Thesis2007(MMPosition) converts the given value of cable length from mm to
%                               encoder counts as seen from the controller

% MM2Counts_Thesis2007.m by Melissa Morris

% ENTER: Length of cable (in millimeters)

% OUTPUT: Position of cable in encoder counts
%**********************************************************************

%*** Variables *******
Dpulley = 21.7;   % mm, Inside diameter of pulley (where cable winds)
WpL = 3;        % Winds per layer, before winds over previous cables
Dcable = 1;     % mm, Diameter of cable
CPR = 5000;     % Counts per rotation (for the controller)
MCL = 274.4;    % mm, Maximum Cable Length

%*** Program *******
CurrentLayer = (MCL-MMPosition)/(WpL*Dpulley);

DistancePerCount = (2*pi/CPR)*((Dpulley/2)+(CurrentLayer*Dcable));
EncoderCountsExact = MMPosition/DistancePerCount;

%Convert to interger so it can become a hex value later
EncoderCounts = round(EncoderCountsExact);  %Return this value
```

```
function CreateBASFile_Thesis2007(FileName)
% CreateBASFile_Thesis2007(SlaveNumber)
%          Creates a .bas file for each slave with all of the header
%          information needed in the file

% CreateBASFile_Thesis2007.m by Melissa Morris

% ENTER: The designation of the slave this is for

% OUTPUT: A file with the header information
%**********************************************************************

%*** Variables *******
```

```
%*** Program *******

SlaveFile = fopen(FileName, 'w');   %Create and overwrite if already there

fprintf(SlaveFile, ''' Slave program for control of the HCTL 1100. \n');
fprintf(SlaveFile, ''' Created by Melissa Morris for thesis project 2007. \n');
fprintf(SlaveFile, ''' ****************************************************************** \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''***** VARIABLES ******* \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''** Output Port ******* \n');
fprintf(SlaveFile, 'symbol READY2GO = 0 \n');
fprintf(SlaveFile, 'symbol STOPPIN = 1 \n');
fprintf(SlaveFile, 'symbol LIMITPIN = 2 \n');
fprintf(SlaveFile, 'symbol OE = 3 \n');
fprintf(SlaveFile, 'symbol CS = 4 \n');
fprintf(SlaveFile, 'symbol ALE = 5 \n');
fprintf(SlaveFile, 'symbol RW = 6 \n');
fprintf(SlaveFile, 'symbol RESET = 7 \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''** Input Port ******* \n');
fprintf(SlaveFile, 'symbol INITPIN = pin2 \n');
fprintf(SlaveFile, 'symbol PROFPIN = pin3 \n');
fprintf(SlaveFile, 'symbol MASTERGOPIN = pin4 \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''** Port C ******* \n');
fprintf(SlaveFile, '''(Used for input/output of addresses and data) \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''** HCTL-1100 Registers ******* \n');
fprintf(SlaveFile, 'symbol FLAGREG = $00 \n');
fprintf(SlaveFile, 'symbol PROGCNTREG = $05 \n');
fprintf(SlaveFile, 'symbol STATUSREG = $07 \n');
fprintf(SlaveFile, 'symbol SAMPLETIMERREG = $0F \n');
fprintf(SlaveFile, 'symbol READACTPOSREGMSB = $12 \n');
fprintf(SlaveFile, 'symbol READACTPOSREG = $13 \n');
fprintf(SlaveFile, 'symbol READACTPOSREGLSB = $14 \n');
fprintf(SlaveFile, 'symbol PRESETACTPOSREGMSB = $15 \n');
fprintf(SlaveFile, 'symbol PRESETACTPOSREG = $16 \n');
fprintf(SlaveFile, 'symbol PRESETACTPOSREGLSB = $17 \n');
fprintf(SlaveFile, 'symbol ZEROFILTERREG = $20 \n');
fprintf(SlaveFile, 'symbol POLEFILTERREG = $21 \n');
fprintf(SlaveFile, 'symbol GAINFILTERREG = $22 \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, ''' * For Setting Up (Position Motion) \n');
fprintf(SlaveFile, 'symbol CMDPOSREGMSB = $0C \n');
fprintf(SlaveFile, 'symbol CMDPOSREG = $0D \n');
fprintf(SlaveFile, 'symbol CMDPOSREGLSB = $0E \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, ''' * For Trapezodial Motion *** \n');
fprintf(SlaveFile, 'symbol FINALPOSREGLSB = $29 \n');
fprintf(SlaveFile, 'symbol FINALPOSREG = $2A \n');
fprintf(SlaveFile, 'symbol FINALPOSREGMSB = $2B \n');
fprintf(SlaveFile, 'symbol ACCELERATIONLSB = $26 \n');
fprintf(SlaveFile, 'symbol ACCELERATIONMSB = $27 \n');
fprintf(SlaveFile, 'symbol MAXVELOREG = $28 \n');
fprintf(SlaveFile, '\n');
```

```
fprintf(SlaveFile, '''** Program Variables ******* \n');
fprintf(SlaveFile, '''Note:  Bits 0 and 1 are reserved for use in program \n');
fprintf(SlaveFile, 'symbol HCTLADDRESS = b2 \n');
fprintf(SlaveFile, 'symbol HCTLDATA = b3 \n');
fprintf(SlaveFile, 'symbol NOWRITETIME = b4 \n');
fprintf(SlaveFile, 'symbol HCTLSAMPLETIME = $FF \n');
fprintf(SlaveFile, 'symbol K_GAIN = 1 \n');
fprintf(SlaveFile, 'symbol A_ZERO = 0 \n');
fprintf(SlaveFile, 'symbol B_POLE = 0 \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''***** MAIN PROGRAM ******* \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, 'Main:   \n');
fprintf(SlaveFile, '        pins = %%01111110      ''Reset starts low \n');
fprintf(SlaveFile, '        wait 1              ''Minimum of 10 microseconds needed for reset \n');
fprintf(SlaveFile, '        pins = %%11111110      ''Set read/write and reset, etc. to high \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = SAMPLETIMERREG \n');
fprintf(SlaveFile, '        let HCTLDATA = HCTLSAMPLETIME \n');
fprintf(SlaveFile, '        gosub WriteData         ''Set the sample timer to the maximum speed
\n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = STATUSREG \n');
fprintf(SlaveFile, '        let HCTLDATA = %%00000001 \n');
fprintf(SlaveFile, '        gosub WriteData         ''Turn on PWM sign reversal inhibit \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = GAINFILTERREG \n');
fprintf(SlaveFile, '        let HCTLDATA = K_GAIN \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write the gain into the controller \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = ZEROFILTERREG \n');
fprintf(SlaveFile, '        let HCTLDATA = A_ZERO \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write the zero value into the controller \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = POLEFILTERREG \n');
fprintf(SlaveFile, '        let HCTLDATA = B_POLE \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write the pole value into the controller \n');
fprintf(SlaveFile, '\n');

fclose(SlaveFile);
```

```
function AddBegin2BASFile_Thesis2007(FileName,StartPos)
% AddBegin2BASFile_Thesis2007(FileName)
%       Adds preset position to .bas files

% AddBegin2BASFile_Thesis2007.m by Melissa Morris

% ENTER: The filename to update, the next position
%   address and the current step being printed

% OUTPUT: Adds current (start) position to .bas file
%**********************************************************************

%*** Variables *******
PosMSB = 0;
PosReg = 0;
```

```
PosLSB = 0;
NextPosHex = 0;

%*** Program *******

%Convert the encoder counts into hex and split by bytes
NextPosHex = dec2hex(StartPos,6);
PosChar1 = NextPosHex(1);
PosChar2 = NextPosHex(2);
PosChar3 = NextPosHex(3);
PosChar4 = NextPosHex(4);
PosChar5 = NextPosHex(5);
PosChar6 = NextPosHex(6);
PosMSB = strcat(PosChar1,PosChar2);
PosReg = strcat(PosChar3,PosChar4);
PosLSB = strcat(PosChar5,PosChar6);

SlaveFile = fopen(FileName, 'a');   %Create and overwrite if already there

fprintf(SlaveFile, '        let HCTLADDRESS = PRESETACTPOSREGMSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosMSB);
fprintf(SlaveFile, '        gosub WriteData         ''Make sure the current position is zero \n');
fprintf(SlaveFile, '        let HCTLADDRESS = PRESETACTPOSREG \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosReg);
fprintf(SlaveFile, '        gosub WriteData         ''Make sure the current position is zero \n');
fprintf(SlaveFile, '        let HCTLADDRESS = PRESETACTPOSREGLSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosLSB);
fprintf(SlaveFile, '        gosub WriteData         ''Make sure the current position is zero \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = CMDPOSREGLSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosLSB);
fprintf(SlaveFile, '        gosub WriteData         ''Write the LSB of the command register (current
location) \n');
fprintf(SlaveFile, '        let HCTLADDRESS = CMDPOSREG \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosReg);
fprintf(SlaveFile, '        gosub WriteData         ''Write to the command register (current
location) \n');
fprintf(SlaveFile, '        let HCTLADDRESS = CMDPOSREGMSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $%s \n',PosMSB);
fprintf(SlaveFile, '        gosub WriteData         ''Write to MSB of the command register (current
location) \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '''**** Entering Position Mode Here ******** \n');
fprintf(SlaveFile, '        let HCTLADDRESS = PROGCNTREG \n');
fprintf(SlaveFile, '        let HCTLDATA = $03 \n');
fprintf(SlaveFile, '        gosub WriteData         ''Enter Position Mode \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = ACCELERATIONMSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $0F \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write acceleration information (part 1) \n');
fprintf(SlaveFile, '        let HCTLADDRESS = ACCELERATIONLSB \n');
fprintf(SlaveFile, '        let HCTLDATA = $FF \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write acceleration information (part 2) \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '        let HCTLADDRESS = MAXVELOREG \n');
fprintf(SlaveFile, '        let HCTLDATA = $19 \n');
fprintf(SlaveFile, '        gosub WriteData         ''Write maximum velocity information \n');
```

```
fprintf(SlaveFile, '\n');

fclose(SlaveFile);
```

```
function AddCmd2BASFile_Thesis2007(FileName, NextPosition, CurrentStep)
% AddCmd2BASFile_Thesis2007(SlaveNumber, NextPosition, CurrentStep)
%         Adds next position to .bas files

% AddCmd2BASFile_Thesis2007.m by Melissa Morris

% ENTER: The filename to update, the next position
%   address and the current step being printed

% OUTPUT: Adds new position to file
%*************************************************************************

%*** Variables *******
PosMSB = 0;
PosReg = 0;
PosLSB = 0;
NextPosHex = 0;

%*** Program *******

%Convert the encoder counts into hex and split by bytes
NextPosHex = dec2hex(NextPosition,6);
PosChar1 = NextPosHex(1);
PosChar2 = NextPosHex(2);
PosChar3 = NextPosHex(3);
PosChar4 = NextPosHex(4);
PosChar5 = NextPosHex(5);
PosChar6 = NextPosHex(6);
PosMSB = strcat(PosChar1,PosChar2);
PosReg = strcat(PosChar3,PosChar4);
PosLSB = strcat(PosChar5,PosChar6);

SlaveFile = fopen(FileName, 'a');   %Create and overwrite if already there

fprintf(SlaveFile,'   high READY2GO \n');
fprintf(SlaveFile,'PrgmStartLoop%d: \n',CurrentStep);
fprintf(SlaveFile,'   if MASTERGOPIN = 0 then PrgmStartLoop%d \n',CurrentStep);
fprintf(SlaveFile,'   low READY2GO \n');
fprintf(SlaveFile,' \n');
fprintf(SlaveFile,'''** NEW POSITION COMMAND ** \n');
fprintf(SlaveFile,'   let HCTLADDRESS = FINALPOSREGMSB \n');
fprintf(SlaveFile,'   let HCTLDATA = $%s \n',PosMSB);
fprintf(SlaveFile,'   gosub WriteData        ''Write destination position info (part 1) \n');
fprintf(SlaveFile,'   pause 1 \n');
fprintf(SlaveFile,'   let HCTLADDRESS = FINALPOSREG \n');
fprintf(SlaveFile,'   let HCTLDATA = $%s \n',PosReg);
fprintf(SlaveFile,'   gosub WriteData        ''Write destination position info (part 2) \n');
fprintf(SlaveFile,'   pause 1 \n');
fprintf(SlaveFile,'   let HCTLADDRESS = FINALPOSREGLSB \n');
fprintf(SlaveFile,'   let HCTLDATA = $%s \n',PosLSB);
fprintf(SlaveFile,'   gosub WriteData        ''Write destination position info (part 3) \n');
fprintf(SlaveFile,'   pause 1 \n');
```

```
fprintf(SlaveFile, '\n');
fprintf(SlaveFile,'   let HCTLADDRESS = FLAGREG   ''Entering Trapezoidal motion here \n');
fprintf(SlaveFile,'   let HCTLDATA = $08 \n');
fprintf(SlaveFile,'   gosub WriteData        ''Begin Trapezoid Motion \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile,'PollTrapPin%d: \n',CurrentStep);
fprintf(SlaveFile,'   if PROFPIN = 1 then goto PollTrapPin%d \n',CurrentStep);
fprintf(SlaveFile, '\n');
```

```
function FinishBASFile_Thesis2007(FileName)
% FinishBASFile_Thesis2007(FileName)
%         Completes the .bas files

% FinishBASFile_Thesis2007.m by Melissa Morris

% ENTER: File name to complete

% OUTPUT: Adds end as well as read and write functions to the file
%**********************************************************************

%*** Variables *******

%*** Program *******

SlaveFile = fopen(FileName, 'a');   %Create and overwrite if already there

fprintf(SlaveFile,'    high READY2GO \n');
fprintf(SlaveFile,'\n');
fprintf(SlaveFile,'    wait 5 \n');
fprintf(SlaveFile,'    end \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile,'''***** SUBROUTINES ******* \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile,'ReadData: \n');
fprintf(SlaveFile,'          dirsc = %%11111111              ''Set port C to export address \n');
fprintf(SlaveFile,'          let HCTLDATA = $00     ''Clear data in HCTLDATA word \n');
fprintf(SlaveFile,'          pinsc = HCTLADDRESS ''Send address to read \n');
fprintf(SlaveFile,'          low ALE \n');
fprintf(SlaveFile,'          high ALE \n');
fprintf(SlaveFile,'          low CS \n');
fprintf(SlaveFile,'          high CS \n');
fprintf(SlaveFile,'          pause 7                ''Pause before OE can go low \n');
fprintf(SlaveFile,'          pinsc = %%00000000             ''Clear output \n');
fprintf(SlaveFile,'          dirsc = %%00000000             ''Set port C to read input \n');
fprintf(SlaveFile,'          low OE \n');
fprintf(SlaveFile,'          let w0 = $00           ''Clear value of w0, since only half will be used
\n');
fprintf(SlaveFile,'          pause 1                ''Insures enough time passes \n');
fprintf(SlaveFile,'          peek 7,b0 \n');
fprintf(SlaveFile,'          high OE \n');
fprintf(SlaveFile,'          let HCTLDATA = b0 \n');
fprintf(SlaveFile,'          pause 1                ''Ensure read cycle is long enough \n');
fprintf(SlaveFile,'          return \n');
fprintf(SlaveFile, '\n');
fprintf(SlaveFile,'WriteData: \n');
```

```
fprintf(SlaveFile,'          dirsc = %%11111111              ''Set port C to export information \n');
fprintf(SlaveFile,'          pinsc = HCTLADDRESS ''Send address to write to \n');
fprintf(SlaveFile,'          low ALE \n');
fprintf(SlaveFile,'          high ALE \n');
fprintf(SlaveFile,'          low CS \n');
fprintf(SlaveFile,'          pinsc = HCTLDATA                ''Write data to the HCTL-1100 \n');
fprintf(SlaveFile,'          low RW \n');
fprintf(SlaveFile,'          high CS \n');
fprintf(SlaveFile,'          pause 1  \n');
fprintf(SlaveFile,'          high RW \n');
fprintf(SlaveFile,'          pause 7                    ''Insure enough time for next cycle \n');
fprintf(SlaveFile,'          pinsc = %%00000000              ''Clear output \n');
fprintf(SlaveFile,'          return \n');
fprintf(SlaveFile, '\n');

fclose(SlaveFile);
```

Master Microcontroller Source Code:

```
' * This program is for running the master controller w/o serial coms
' *  It also implements a user interface
' * It plays whatever is loaded in slaves – not necessary a circle, just change steps
' *  This is currently a 40X PICAXE
' * Created by: Melissa Morris for Thesis Project 2007
' *******************************************************************

'***** VARIABLES *******

'** Output Port *******
symbol SPEAKER = 0
symbol SLAVE1GO = 1
symbol SLAVE2GO = 2
symbol SLAVE3GO = 3
symbol LEDRUN = 4
symbol LEDREADY = 5
symbol LEDSTOP = 6
symbol LCDCOM = 7

'** Input Port *******
symbol HANDLEBUTTON = pin0
symbol SLAVE1READYPIN = pin1
symbol SLAVE2READYPIN = pin2
symbol SLAVE3READYPIN = pin3
symbol GOBUTTON = pin4
symbol STOPBUTTON = pin5

'** Port C ******* (Note: Need to specifiy "portc" with command)

'** Program Variables *******
'Note:  Bits 0 and 1 are reserved for use in program
symbol TIMESTHRU = b2
symbol FINISHEDCHECK = b3
```

```
symbol STEPNUMBER = b4

'***** MAIN PROGRAM *******

Main:
        pins = %00000000        'Start with all low
        high LEDRUN
        wait 1              'Let other microcontroller get going
        low LEDRUN
        high LEDREADY           'Signal that ready to go

        sound SPEAKER, (100,75)
        serout LCDCOM,n2400,("Hello There!")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("  Welcome!")
        STEPNUMBER = 0
        wait 2

StartUpMenu:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Make a selection")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Y:Play N:Other")
ModeMenu1:
                                'Note: if both are pressed, stop is always read first
        if STOPBUTTON = 1 and GOBUTTON = 0 then SecondMenu
        if GOBUTTON = 1 and STOPBUTTON = 0 then PlaySelectRoutine
        pause 100               'If no button, wait for bounce and check again
        goto ModeMenu1

SecondMenu:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Sorry, no more")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Y:Play N:Quit")
ModeMenu2:
        if STOPBUTTON = 1 and GOBUTTON = 0 then ShutDownRoutine
        if GOBUTTON = 1 and STOPBUTTON = 0 then PlaySelectRoutine
        pause 100               'If no button, wait for bounce and check again
        goto ModeMenu2

ShutDownRoutine:
        '*** turn off controllers
        low LEDRUN
        low LEDREADY
        high LEDSTOP
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Goodbye!")
        wait 3
        sound SPEAKER,(50,50)
        serout LCDCOM,n2400,(254,8)
        low LEDSTOP
        wait 2
        end

'***** SUBROUTINES *********
```

```
interrupt:                    'only works for circle
        pause 150
                if HANDLEBUTTON = 1 then
                        setint %00000000, %00000001
                        return
                else
                        'if not, then the grip was lost for too long
                        '**** E-stop (and routine)
                        high LEDSTOP
                        low LEDRUN
                        sound SPEAKER, (100,50,50,75)
                        serout LCDCOM,n2400,(254,1)
                        serout LCDCOM,n2400,("STOPPED- No Grip")
                        serout LCDCOM,n2400,(254,192)
                        serout LCDCOM,n2400,("Y:Continue N:End")
StopMenu1:
                        if STOPBUTTON = 1 and GOBUTTON = 0 then ShutDownRoutine
                        if GOBUTTON = 1 and STOPBUTTON = 0 then ContRun
                        pause 100               'If no button, wait for bounce and check again
                        goto StopMenu1
                endif
        return

ContRun:
        setint %00000000, %00000001
        return

FaultRoutine1:
'One or more slaves are not ready to get data before time-out
        pins = %00000000
        sertxd("Problem with one or more slaves - not ready for signal! ")
        if SLAVE1READYPIN = 0 then Msg1
        if SLAVE2READYPIN = 0 then Msg2
        if SLAVE3READYPIN = 0 then Msg3
        wait 60
        end

FaultRoutine2:
'One or more slaves are not gettting to their point within time
        pins = %00000000
        '***** E-stop all motors!
        sertxd("Position not being reached in time! ")
        if SLAVE1READYPIN = 1 then Msg1
        if SLAVE2READYPIN = 1 then Msg2
        if SLAVE3READYPIN = 1 then Msg3
        wait 60
        end

PlaySelectRoutine:
 PlayMenu1:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Which to run?")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Y:Triangle N:ext")
 PlaySelect1:
        if STOPBUTTON = 1 and GOBUTTON = 0 then PlayMenu2
        if GOBUTTON = 1 and STOPBUTTON = 0 then CircleRunRoutine
```

```
        pause 100                    'If no button, wait for bounce and check again
        goto PlaySelect1
 PlayMenu2:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Which to run?")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Y:Circle N:Other")
 PlaySelect2:
        if STOPBUTTON = 1 and GOBUTTON = 0 then PlayMenu3
        if GOBUTTON = 1 and STOPBUTTON = 0 then SquareRunRoutine
        pause 100                    'If no button, wait for bounce and check again
        goto PlaySelect2
 PlayMenu3:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Which to run?")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Y:Other N:Exit")
 PlaySelect3:
        if STOPBUTTON = 1 and GOBUTTON = 0 then StartUpMenu
        if GOBUTTON = 1 and STOPBUTTON = 0 then PlayMenu1
        pause 100                    'If no button, wait for bounce and check again
        goto PlaySelect3

'******************** Circle Routine
CircleRunRoutine:
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Grip Handle")
        serout LCDCOM,n2400,(254,192)
        serout LCDCOM,n2400,("Get Ready")
 HandleGripCheckLoop:
        if STOPBUTTON = 1 then StartUpMenu
        if HANDLEBUTTON = 0 then HandleGripCheckLoop

        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Starting...")
        high LEDRUN
        low LEDREADY
        sound SPEAKER, (50,50,100,75,30,100)
        setint %00000000, %00000001  'Interrupt if joystick is released or stop pressed

 NextPosCmdLoop:
        TIMESTHRU = 0
        FINISHEDCHECK = 0
 PosCmdLoop:
        TIMESTHRU = TIMESTHRU + 1
        if TIMESTHRU = 500 then FaultRoutine1
        if SLAVE1READYPIN = 0 or SLAVE2READYPIN = 0 or SLAVE3READYPIN = 0 then
PosCmdLoop
        serout LCDCOM,n2400,(254,1)
        serout LCDCOM,n2400,("Running")
        high LEDRUN
        sertxd("The slaves are all ready! ")
        pins = %00001110
        pause 5            'Make sure all slaves can see the high
        pins = %00000000
        STEPNUMBER = STEPNUMBER + 1
 SlaveRunCheckLoop:
```

```
	FINISHEDCHECK = FINISHEDCHECK + 1
	if FINISHEDCHECK = 1000 then FaultRoutine2
	if SLAVE1READYPIN = 1 or SLAVE2READYPIN = 1 or SLAVE3READYPIN = 1 then
SlaveRunCheckLoop
	sertxd("All slaves are running. ")
	pins = %00000000
	if STEPNUMBER = 33 then CircleEnd
	goto NextPosCmdLoop

CircleEnd:
	setint %00000000, %00000000
	pause 500
	high LEDSTOP
	low LEDRUN
	sound SPEAKER, (100,50,50,75)
	serout LCDCOM,n2400,(254,1)
	serout LCDCOM,n2400,("Program Complete")
	serout LCDCOM,n2400,(254,192)
	serout LCDCOM,n2400,("  One Moment...")
	high LEDREADY
	low LEDSTOP
	goto StartUpMenu

SquareRunRoutine:
	serout LCDCOM,n2400,(254,1)
	serout LCDCOM,n2400,("Sorry, You only")
	serout LCDCOM,n2400,(254,192)
	serout LCDCOM,n2400,("get a triangle!")
	wait 1
	goto CircleRunRoutine

'*** Messages **************************

Msg1:
	sertxd("Slave 1 is not ready. ")
	serout LCDCOM,n2400,(254,1)
		serout LCDCOM,n2400,("STOPPED!!")
		serout LCDCOM,n2400,(254,192)
		serout LCDCOM,n2400,("Cont 1 Error")
	return

Msg2:
	sertxd("Slave 2 is not ready. ")
		serout LCDCOM,n2400,(254,1)
		serout LCDCOM,n2400,("STOPPED!!")
		serout LCDCOM,n2400,(254,192)
		serout LCDCOM,n2400,("Cont 2 Error")
	return

Msg3:
	sertxd("Slave 3 is not ready. ")
		serout LCDCOM,n2400,(254,1)
		serout LCDCOM,n2400,("STOPPED!!")
		serout LCDCOM,n2400,(254,192)
		serout LCDCOM,n2400,("Cont 3 Error")
	return
```