Thesis Outline

Nicholas Berezny

July 11, 2019

Chapter 1

Literature Review and Preliminary Research

1.1 Stroke and Rehabilitation

Stroke is a cerebrovascular disease which effects approximately 62,000 Canadians and 795,000 Americans each year [1, 2]. It is among the leading casuses of adult disability [3], with some form of motor impairment effecting 80% of patients, which can result in limited mobility and muscle control [4]. Ischaemic stroke is the most common form of stroke, accounting for around 80% of all cases [5]. It is caused by blood vessel occlusion, either directly in the brain (thrombotic), or due to the migration of a clot formed somewhere else in the body (embolic). The subsequent lack of oxygen causes the death of brain tissue and neurological deficits [6], which manifest as functional impairments in activities associated with the affected brain region. Stroke complications can include hemiparesis (weakness on one side of the body), muscle spacitity, loss of motor control, and loss of dexterity, all of which can effect the patient's ability to perform activities of daily living (ADL's) and compromise their autonomy. In addition to medical intervention in the form of pharmaceuticals and surgery, the stroke patient may require rehabilitation to reduce functional impairment [7]. It is imperative to administer treatment during the acute phase of the stroke (*i.e.* within days of the stroke) [6].

Older treatment techniques relied on compensatory measures to assist stroke patients in regaining autonomy [8]. Using rehabilitation to restore autonomy began to be emphasized once the concept of neural plasticity became more widely accepted. Neural plasticity is the process by which new connections are formed in the central nervous system in response to experience or activities performed by the subject [9]. When a stroke patient repetitively performs motor tasks, new neural connections in the brain are formed which take on the role of the regions of the brain which were damaged. This allows the patient to re-learn tasks such as walking by going through intensive rehabilitation.

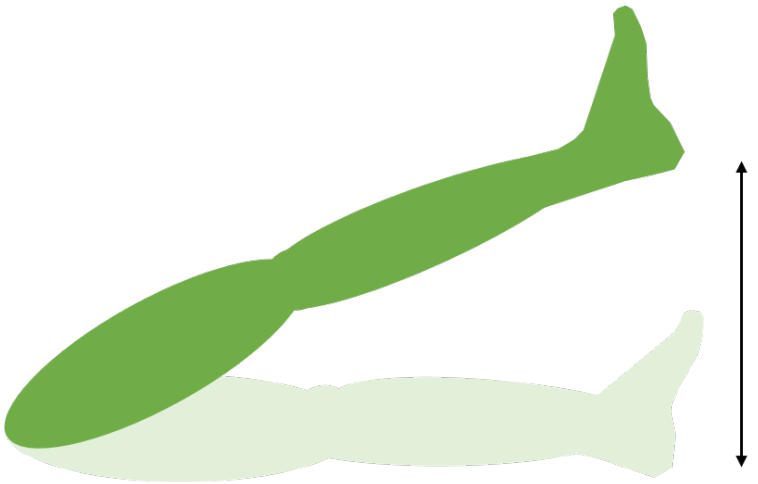

Figure 1.1: Example of the leg lift exercise

Stroke rehabilitation encompasses a broad range of therapeutic activities, typically carried out by occupational- and physiotherapists. Physiotherapists (PT's) tend to focus on gross motor movements in the upper and lower limbs, while occupational therapists (OT's) focus on fine motor control, dexterity, and specific ADL's. These two goals are inextricably linked, and so both PT's and OT's form a team which works together to improve patient outcomes. The rehabilitation carried out by both OT's and PT's involves the repetition of a target movement or task. Depending on the abilities of the patient, the therapist may provide support or assistance. For example, lower-limb rehabilitation begins with simple legs movement performed in bed, such as leg lifts (Fig. 1.1), knee flexion/extension (Fig. 1.2), hip abductions, *etc* [include pictures]. The therapist can either assist the patients leg

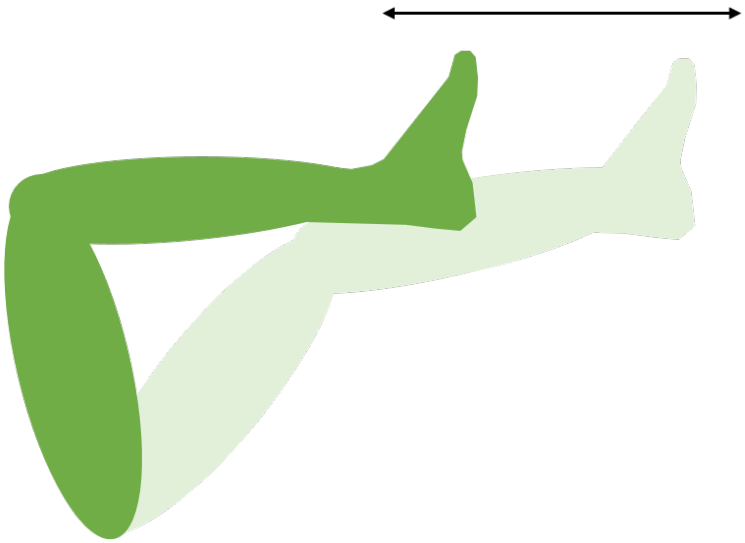

Figure 1.2: Example of the knee extension/flexion exercise

through the movement, or allow patients to perform the movement on their own, or even provide light resistance if the patient is more advanced. Next, the patient will practice sitting upright, going from sit to stand, and practice bearing weight and balancing while standing. Eventually the patients will be ready for more complex tasks with assistance from the therapist, including walking, heel strike, and stair climbing. The patient may advance to the point where they can walk without therapist assistance, using either parallel bars, a walker, or potentially without any assistance. The stages of lower-limb rehabilitation are summarized below:

1. Bed-bound exercises with assistance
2. Sit-to-stand and weight bearing
3. Assisted gait therapy (with therapist moving the leg through the gait trajectory)
4. Other out-of-bed activities, such as heel strike, stair climbing, etc.
5. Walking without therapist assistance (using parallel bars or a walker)

1.1.1 Rehabilitation Guidelines

Canadian stroke best practices recommend that therapy involves the repetition of intensive, task-specific activities in a complex and stimulating environment [2]. Lower-limb rehabilitation should be goal-orientated towards selected tasks such as walking and sit-to-stand. Robotics, virtual reality, and biofeedback are all mentioned as potential adjuncts, but are not recommended to replace traditional therapy. It is also recommended that patient's should receive rehabilitation as early as possible once they are deemed medically ready. Early mobilisation is recommended (within 24-48 hours of stroke, [10], but not before 24 hours due to findings that this actually reduces functional outcomes, [11]). Active participation on the part of the patient is also imperative [4], as it has been found that this facilitates neural plasticity. This requires that the patient be engaged in the therapy (hence the need for stimulating environments), and also requires the therapist to ensures that the rehabilitation is not passive.

Therapy dosage also has an effect on functional outcomes. For example, one study found positive correlation between time scheduled for therapy and most functional outcomes [12]. Another study sought to determine if current doses of rehabilitation are enough to trigger functional improvements by comparing with animal models [13], and found that the amount of rehabilitation they observed was inadequate. Another study found that stroke inpatients spend on average 13% engaged in physical activities and 5.2% of their time with a therapist [14]. Other studies confirm that stroke patients spend a significant amount of time in or sitting near their bed [15]. These results indicate the importance of scheduling time specifically for rehabilitation for stroke inpatients.

Other factors may restrict the PT's and OT's ability to accomplish the recommended best practices. The intensity and duration of the therapy is limited both by the endurance of the patient and of the therapist, as the physical burden of manually assisting the patient may be high [16]. Furthermore, providing task-specific and stimulating therapy is difficult, particularly when doing bed-bound exercises which are not easily relatable to ADL's. One study found that therapy dosage may be affected by understaffed units, resulting in less

rehabilitation time with therapists [17].

1.2 Robotic Rehabilitation

Rehabilitation robots have been introduced to improve the stroke rehabilitation process by automating the physical assistance usually supplied by therapists. Rehabilitation requires the repetition of a precise motion, a task well-suited to robotics. Over the past few decades, a variety of robotic platforms have been developed for many different therapeutic activities, from lower-limb gait training to fine motor skills in the hand. There are many other potential benefits for using robotics for rehabilitation, including the following.

- **Relieving Physical Burden**: Using robots to provide the physical support and assistance required for stroke therapy will relieve the physical burden from therapists, removing the chance of injury or fatigue.

- **Increased Intensity and Duration**: Both the intensity and duration of a therapy session may be limited when relying on a therapist to provide physical assistance. With a robot, duration and intensity would only be limited by the patient.

- **Increased Dosage**: Robots could also increase the amount of rehabilitation given to patients. For example, a single therapist could supervise several patient's each using a robotic device, therefore increasing the number of patients a therapist can manage.

- **Providing Performance Measures**: Robots record a number of metrics not available when using traditional therapy, such as velocity, force, trajectory error, and effort. Tracking these measures could allow the therapist to better administrate therapy and track patient improvement.

- **Virtual Environments**: Patient engagement could be increased by incorporating virtual reality, games, and other forms of visual feedback. The robot would serve as the control input to the environment. The relevancy of the therapy could also be improved

by simulating ADL's. Finally, haptic feedback could further immerse the patient in the virtual environment by rendering virtual forces to the patient (see section 1.2.4).

Rehabilitation robots can be broadly categorized as targeting the lower-limb or the upper-limb. Upper-limb robots tend to be lighter weight since they encounter much smaller loads, while lower-limb devices are heavier as they need to support either the weight of the leg or of the entire body. Lower-limb robots can be further categorized as either targeting standing activities such as treadmill walking, or lying/sitting activities [18]. A final subdivision exists between endplate based devices and exoskeletal devices. Endplate-based devices interact with the patient at a single point, usually the end of the limb (the hand or foot), and only apply forces through this point. Exoskeletal devices attach to the user at multiple points, offering greater control over the overall motion of the limb, although they are often more restrictive, more complex, and potentially less safe [19]. One study found evidence that endplate-based devices may actually yield better outcomes than exoskeletal devices, although the comparison was done via meta-analysis of many different studies with significantly different methodologies [20] .

Rehabilitation robots have not yet been fully endorsed by stroke best practices. For example, Canadian best practices list robots as a potential benefit, but do not recommend they replace traditional therapy [2]. This is largely due to the limited evidence for outcomes of robotic rehabilitation. Many studies are small scale case studies, and the few large clinical trials have found conflicting results. One study following 63 patients compared using the Lokomat (see Sec. 1.2.1) to conventional therapy, and found that the robot was less effective [21]. There are many challenges in studying the efficacy of robots in a clinical setting. It is unethical to replace traditional therapy with an unproven method, so typically extra therapy is added to the subject's regimen either from a robot or from a therapist (control). Since most robots are designed to replace time spent in conventional therapy, the robot must consistently be shown to be at least as effective as a therapist at delivering rehabilitation. Furthermore, since most robotic devices are substantially different, results from the study of one lower-limb robot may not apply to another robot. All of this makes "proving" the

efficacy of a rehabilitation robot difficult. That said, there have been many different robotic devices successfully tested, with some even being available commercially.

1.2.1 Robots for Overground Walking

One of the most important activities for lower-limb rehabilitation is gait training, wherein the patient walks with assistance from the therapist. Rehabilitation robots were introduced to provide this assistance by attaching to the patient's leg and guiding them through the correct gait trajectories. This can be accomplished bv an exoskeletal orthosis attached around the knee joint, or by and endplate device which supports and guides the feet.

The Lokomat, ALEX [22], LOPES [23] and ReoAmbulator are examples exoskeletal gait trainers. Each comprise of a powered exoskeleton which can assist the leg through gait trajectories. They all work in conjunction with body-weight supported treadmill training (BWSTT), using a harness to support the patient and a treadmill for walking. The Lokomat and LOPES use impedance control to deliver programmable levels of assistance to the user. ALEX uses a force-field scheme to create an "assist-as-needed" controller which ensures that the user contributes to the movement.

The Lokomat is one of more widely researched platforms. One recent study found that the Lokomat was more effective than conventional gait therapy with regards to certain outcome measures [24], although another found that it was less effective [21]. Despite conflicting results, the Lokomat continues to show promise. It is commercially available – 872 devices can be found worldwide according to their website (as of 2019).

The Haptic Walker [25], G-EO [26], and Gait Trainer [27] are examples of end-plate based rehabilitation robots. Both use actuated platforms under the feet to move the patient through gait trajectories, and a harness to support the patient. The Haptic Walker can perform walking simulations and more complex situations like stumbling or climbing stairs using a 3 DOF system. As can be seen in Figure 1.4, the system is much larger than the exoskeletal systems. The GE-O system is similar in that it can simulate walking and stair climbing, with one study finding that electromyography (EMG) readings were similar

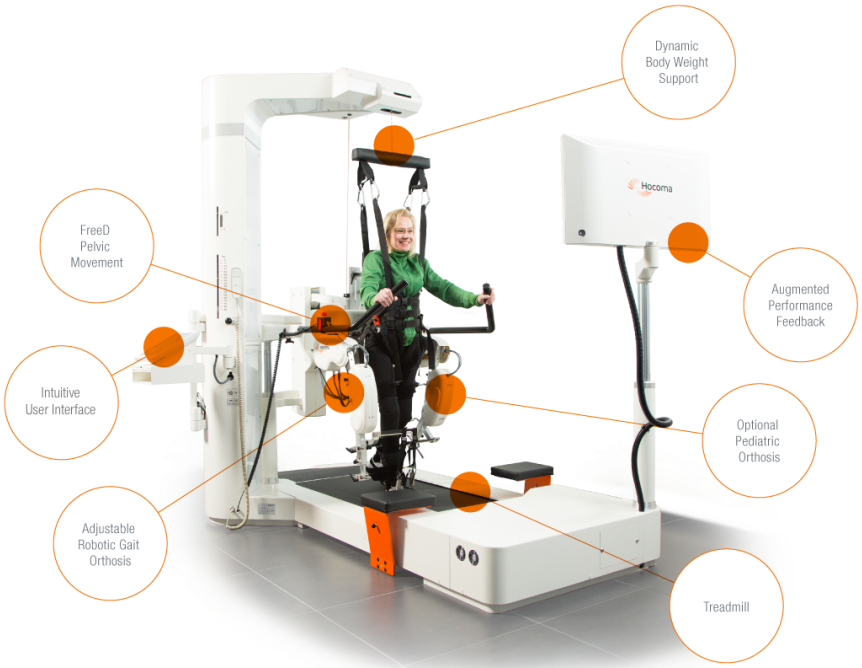

Figure 1.3: The Lokomat (by Hocoma), an exoskeletal robot

between simulated walking with the robot and real walking [26].

Overground walking robots show promise for improving gait rehabilitation. However, these only target a single type of lower-limb rehabilitation. Patients must be somewhat mobile (enough to leave their beds and their rooms) to access these devices, and the complexity and setup require that a therapist be present. Hence these robots are designed to replace time spent on conventional therapy, which makes proving consistent performance difficult as was previously mentioned.

1.2.2 Robots for Sitting or Bed-bound Therapy

Robots for assisting in activities while the patient sits or lies down are less common in the literature. They can be used on acute patients who are less mobile. The MotionMaker, Lambda, and ViGRR are some examples.

The MotionMaker consists of a reclinable chair and two orthoses, which attach to the

Figure 1.4: The Hapticwalker, an endplate based robot

leg only at the foot [28]. It uses a regulator with force feedback to control the exercise. It is designed to be used in conjunction with functional electrical stimulation for spinal cord injury patients, but it's application extends to lower-limb rehabilitation in general.

The Lambda is robot developed for general lower-limb rehabilitation and for fitness purposesd [29]. It uses parallel linkages with a total of 3 DOFs to achieve a variety of motions. An adjustable seat allows the user to recline as needed. Recent developments include the addition of virtual reality and games to engage the patient, enhanced feedback regarding the patients progress, and multiple types of exercises.

The precursor to this project is the Virtual Gait Rehabilitation Robot (ViGRR), a device developed at Carleton University's Advanced Biomechatronics and Locomotion Laboratory [30, 31]. It is a 4-DOF end-plate based robot which can assist a reclined patient move through trajectories in the sagittal plane. The user's foot interfaces with a footplate which is

Figure 1.5: The MotionMaker

magnetically attached to the robot. The magnet can be released if triggered by safety system, including high force, high velocity, or pressing an emergency stop button. An admittance controller is used to provide assistance-as-needed, roughly in proportion to the trajectory error. These desired trajectories can include gait trajectories, linear motion, circular motion, or any other path in the sagittal plane. To increase user engagement, a virtual environment was created which can display visual feedback regarding the desired trajectory, or games such as pushing a block or brick breaker. User's can be further immersed in the virtual environment through haptic feedback. For example, when pushing on virtual wall, the robot will apply a force to the user's foot emulating the wall reaction force.

1.2.3 Controls

The type of controller used is especially important for devices interacting with a human (or any unknown environment), as it will determine the interaction behaviour and stability. The behaviour of the interaction should be optimized to ensure the patient engages in meaningful

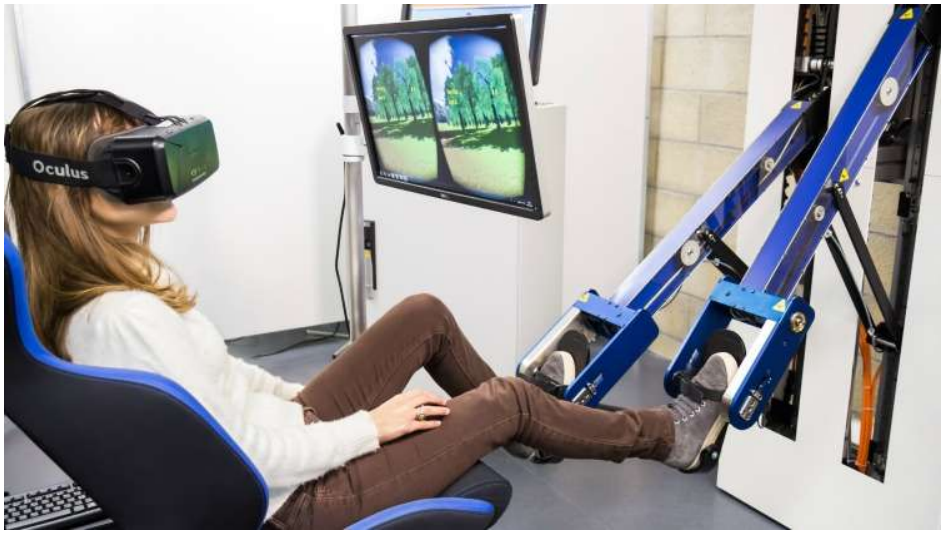

Figure 1.6: The Lambda

therapy driving neural plasticity and recovery. Position control through the desired trajectory is not adequate, as the robot will effectively force the limb through the motion, thus becoming a passive motion machine with no necessary engagement from the patient. Neural plasticity requires engagement, *i.e.* the patient needs to contribute to some extent to the movement of their limbs. Therefore, rehabilitation robotics should only give some assistance to the patient, instead of leading the motion.

Three popular control laws for rehabilitation robotics are listed in [32]:

- **Impedance and admittance** control were introduced by Hogan in [33], and have been the foundation of most interaction controllers since. These controllers determine the dynamic interaction between robot and environment instead of regulating position or force. In this way, an impedance controller can be used to give a programmable amount of assistance to the user through a trajectory, which can be varied by changing the parameters so that an optimal amount of assistance can be obtained for each individual. The desired impedance or admittance is defined in terms of a mass-spring-damper system. Therefore, changing the spring gain will change the level of assistance. The damper gain can be used to resist motion if the patient is more advanced or if strength training is the goal.

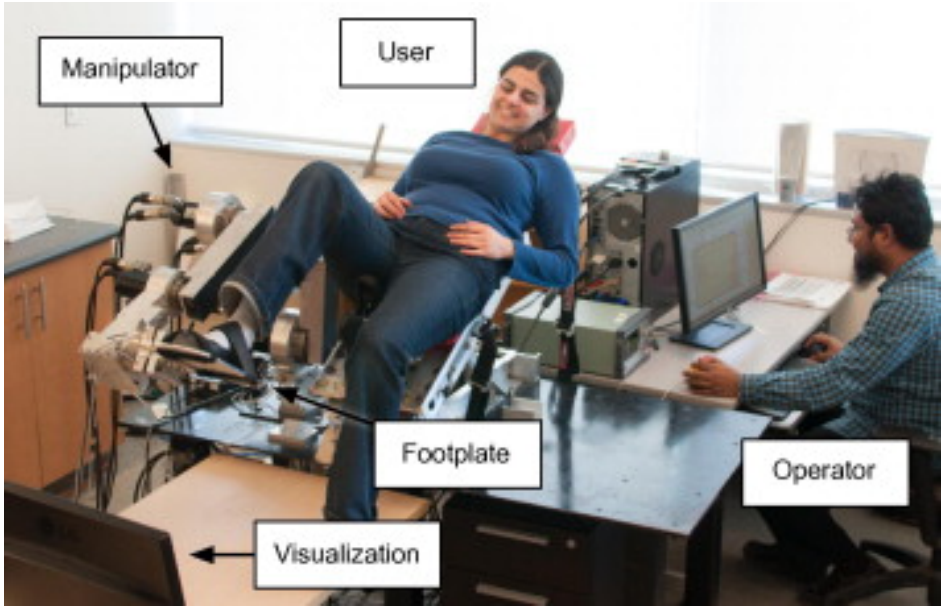

Figure 1.7: ViGRR

- **EMG-based** control uses electromyography (EMG) to measure muscle activity in the patient. The robot then only provides assistance when the intent to move is detected (*i.e.* the patient flexes their muscle). This ensure that the patient is actively engaged in the therapy.

- **Adaptive** control here refers to a controls which changes its behaviour over time based on some predefined law. For example, the amount of assistance supplied by an admittance controller can be changed based on patient performance. Some robots use information from movements in the patient's healthy limb to calculate the optimal control path for the affected limb. Others use assist-as-needed control schemes to only provide assistance when the patient requires it. Artificial Neural Networks have been used to determine when the assistance is needed in robots such as the ALEX and LOPES.

1.2.4 Virtual Environments

Virtual environments or virtual reality refer to digitally created simulations or visualizations, typically displayed through headsets or on computer monitors. It combines visuals, auditory stimulation, and haptic feedback to create an immersive experience. Virtual reality has been introduced into rehabilitation for two primary reasons [34]:

- **Increase Relevancy**: simulate ADL's and real world activities to introduce context into the exercise, and better prepare the patient for these activities outside of the hospital environment.
- **Increase Engagement**: use games and other visuals to make the therapy more exciting and engaging, encouraging more active participation, higher intensity, and higher doses of activity.

Virtual reality can be coupled with rehabilitation robotics to further increase immersion. The robot can be used as a control input to the virtual environment, and the robot can in turn deliver haptic feedback to the user. Haptic feedback allows users to interact directly with the virtual world, for example when walking, the ground reaction force can be applied to the user's foot. This increases the realism of ADL simulations.

A metastudy found evidence that VR may improve functional outcomes over conventional therapy, but admits that most studies are small and prone to bias [34]. One platform used an upper-limb exoskeletal device to apply assistive forces [35]. Another study used a Nintendo Wii gaming system to promote therapy, and found it to be a feasible and safe method [36]. Another study used the Phantom Omni, a pen attached to a robotic arm that can deliver haptic feedback from a virtual environment [37]. A serious gaming approach is used in [38], where the Unity3D engine was used to create a virtual world and a MS Kinetic sensor to detect the user's motions.

1.3 Interviews and Shadowing

The rise of the human-centred design paradigm underscores the importance of involving end users in the design process, especially when the designers lack expertise in the field of interest. This is certainly the case for this project as none of the engineers involved have ever delivered stroke therapy. As such, the first step of the design process was to consult the end user and collect qualitative data and advice. In the case of rehabilitation robots, there are two types of end-users: the stroke patient using the device, and the therapists who decide when and how to incorporate the device into their rehabilitation regimen. The therapist, however, has greater expertise with stroke rehabilitation in general, and will ultimately be the one who decides the device's fate on the stroke ward. Therefore, it was decided to interview and shadow several PT's and OT's concerning the direction of the project. Patient's were not consulted, due to their lack of expertise in stroke rehabilitation, although their input will be vital in future evaluation experiments.

Fieldwork was conducted at the Civic campus of the Ottawa Hospital (TOH), in cooperation with Dr. Dariush Dowlatshahi, and with ethics approval from the Ottawa Health Science Network Research Ethics Board (OHSN-REB). Three PT's and two OT's were recruited for two days of interviews and shadowing. Inclusion criteria included experience working on the stroke ward delivering rehabilitation to stroke patients. Not enough data was collected to perform any meaningful statistic – instead, key insights guided the fundamental concept of the device, and information was gathered not otherwise available through a literature review. Some topics of interest included:

- Typical acute, bed-bound rehabilitation
- Patient Engagement – how the therapists keep patients motivated and interactive
- The hospital environment
- The therapists' views on rehabilitation robotics

1.3.1 Typical Rehabilitation

Bed-bound lower-limb rehabilitation is one of the first therapeutic activities administered at the stroke ward, and it continues to be used even after patients are mobile. Bed-bound exercises tend to be simple movements, including knee flexion/extension, leg lifts, hip adduction/abduction, and ankle rolls. All exercises involve the therapist manually manipulating the patient's leg, providing assistance to whatever extent the therapist deems necessary. If the patients are advanced, the therapist may choose to resist motion instead of assisting.

Parameters from typical rehabilitation will help determine the requirements of the new robotic system. Therapists estimated that they exert a maximum of 10 - 15 lbs of force to the leg. The range of motion of patients varied, but many could go through the full range of the leg. The exercises were done slowly at a consistent speed. The role of the therapist can be summarized as providing support to leg and providing assistance through the motion.

Some patient's are capable of performing bed-bound exercises independent of the therapist. Performing independent practice can increase total rehabilitation dosage and is recommended by stroke guidelines [2]. However, some patients who are either at an earlier stage of recovery or have suffered more severe strokes are incapable of performing bed-bound exercises without assistance, and so their total dosage is limited to time spent with the therapist.

1.3.2 Patient Engagement

It is important for the patient to be actively engaged in the exercise. This is accomplished through verbal encouragement, and by requiring the patient to initiate the movement. Therapists also used physical cues to guide the patient, for example by tapping the leg if it needs adjusting. Many of the patients required verbal or physical engagement throughout the process. This is in part due to a phenomenon known as neglect, wherein the stroke victim has trouble focusing on the effected side of the body. Sometimes, pictures and posters (*e.g.*of cats and dogs) were posted in front of the exercise machines in the therapy room. Another common practice is to switch to a new exercise if the patient is showing signs of losing focus.

1.3.3 Hospital Environment

Space in the hospital is limited. Many rooms house four patients, and are often crowded by hospital staff, family, and equipment. The beds themselves vary in model and size, but all have some common characteristics: movable guards on the side, removable baseboard, and a tiltable frame controlled from a panel. There appear to only be outlets behind the beds near the floor, but this will again vary from room to room. There is also a therapy room where more advanced patients go to practice sit-to-stand, walking, stair-climbing, etc. This room is more spacious and also includes beds (albeit simpler beds that cannot be tilted and that do no have guards).

1.3.4 Therapy Schedule

Therapists generally have a high workload and so must carefully schedule time with patients. At this particular hospital, patients received on average 30 minutes every other day with a physiotherapist for lower-limb rehabilitation. Most patients were fatigued by the end of the session, and so increasing the duration beyond 30 minutes is not feasible. However, the therapists recognized that the patients could benefit from more sessions throughout the week.

1.3.5 Other Comments and Concerns

Most therapists had concerns over safety. They recognized that the robot could potential apply forces which could harm the patient. One point of concern was rotation of the leg out of position such that the force was applied incorrectly. For example, during the knee flexion/extension exercise, if the leg were to rotate out of the sagittal plane, the robot could apply a torque about the hip which would bend the leg further and potentially injure the patient. Therapists suggested having multiple support points along the leg to keep the leg in position.

Other comments included the need for comfort, particularly using comfortable material

where the robot and human interact. They also mentioned the use of tactile methods of engaging the patient (*e.g.* tapping or vibrations applied to the leg). Some indicated that combining upper and lower-limb activities could be beneficial (*e.g.* by using a joystick with the robot).

1.4 Device Concept

With ViGRR, our goal was to design a lower-limb rehabilitation robot targeting acute patients by creating a device compatible with a patient who is lying or reclined in a hospital bed. ViGRR accomplishes this, and has been validated on healthy subjects. However, some aspects of ViGRR prevent it from being used easily in the hospital, including its high weight, high power, and size. The next stage in the project is to repackage ViGRR into a more accessible and light-weight design.

This new prototype should align with ViGRR's fundamental concept: a lower-limb robot capable of assisting bed-bound acute stroke patients with rehabilitation. The device should be a viable tool for therapists within the hospital to use to increase functional outcomes for their patients. It should replicate the role of the therapist when providing assistance for bed bound exercises. There are three primary roles:

1. Support Leg
2. Provide Assistance through the motion
3. Engage the patient

Supporting the leg will be accomplished through the physical structure of the device. Assistance will be provided using force feedback and interaction controllers which can ensure that the patient is contributing to some extent. The patient will be engaged using virtual environments and haptic feedback.

Bed-bound exercises include a variety of simple movements, such as leg lifts, knee extensions, and hip abductions. Since this device is a proof-of-concept, it was decided that

we would initially target a single exercise, evaluate its performance in the hospital, and then extend to target other exercises by adding degrees-of-freedom. We chose the knee flexion/extension exercise for the following reasons:

- It is a commonly used bed-bound exercise
- It is important due to its relation to the sit-to-stand activity, which is the first stage of getting the patient mobile

The knee flexion/extension exercise requires just a single horizontal degree-of-freedom. The therapist uses one hand to support the leg under the knee, and the other hand to support the foot and apply force through the heel. They slowly move the leg back and forth, encouraging the patient to push. They are careful to keep the leg aligned in the sagittal plane.

There are several use cases for the device. The therapist could use it during therapy instead of manually moving the legs. This would relieve them of the physical burden, the virtual environments may encourage the patient to participate more, and the data collected could be used to track progress. The therapist could use multiple devices simultaneously to provide care to multiple patients. This would allow them to see more patients and thus increase therapy dosage. Finally, since the device replicates the role of the therapist, the device could be used separately from the therapist, under the supervision of other hospital staff. This would encourage patient's to practice independently, thereby increasing therapy dosage.

In addition to these requirements, the device should be usable within a hospital environment. The device should be safe such that the patient is not harmed, and should also be perceived as safe so that the therapists and patients are willing to use it. The device should fit on a hospital bed, and should be lightweight and small enough to be manageable within a hospital room. The set-up of the device should be quick and easy.

1.5 Contributions and Outline

A one DOF robot was designed and created, consisting of a DC motor, force sensor, belt drive, footplate, frame, and a number of smaller components. A printed circuit board (PCB) was created to route signals, do preprocessing, and distribute power. System software was implemented on a real-time linux system. A controller was created in C, which can operate the device using admittance control, haptic feedback, a basic physics engine, communication with a UI server, and data logging. Real time functionality is ensured using POSIX standards for memory locking, multi-threading, etc. The UI is implemented as website using the React framework, and some simple games are created using the THREE.js 3D web graphics library. Preliminary experiments are run, including functionality tests and healthy subject testing.

- Chapter 2 covers the design of the device. This includes preliminary design choices, the design of the physical structure, the actuation and sensors, the design of the electronics (power distribution, signal routing and preprocessing), and the chosen computer hardware.

- Chapter 3 covers the controller and its implementation in the software, including the basic admittance control, and the haptic feedback with physics engine. The design of the user interface (UI) and some basic games/visualisations are also discussed. The communication and the overall software design within real-time Linux is explained, including the how the modular design will allow it to be easily adapted to future design iterations.

- Chapter 4 covers testing and experiments, including functional testing of the software, controller, and UI. The methods and results from a small study on healthy subjects are presented, with questionnaire answers regarding the subjective experience of participants, along with quantitative performance metrics.

- Chapter 5 concludes with a summary of contributions and recommendations for future work. This includes features that were required which have not yet been implemented due to time constraints, and ideas for future direction of the project.

Chapter 2

Device Design and Implementation

2.1 Design Iterations

This sections lists the early design decisions and iterations, along with the justification. There were three significant design decisions to be made:

1. The number of degrees of freedom (DOF) in the robot. The final decision was to use one linear DOF.

2. The type of linear actuator to use. Options included pneumatics, ball screws, and belt drives; the final design uses a belt drive.

3. The robots placement within the room (on the bed v.s. beside the bed). The final design sits on the bed.

4. The conceptual design of a secondary leg support which would prevent the leg from rotating out of position. This was never realised, however the design is presented here for future work. It consists of a 3 DOF passive linkage which can follow the leg through the motion and provide lateral support.

5. The controller can be run from either a microcontroller or a full computer. Due to the plans to increase complexity in the future, as well as the need for a user interface, a computer was chosen.

2.1.1 Degrees of Freedom

This project is a continuation of ViGRR, a 4 DOF planar robot. The goal is to repackage ViGRR in a smaller form more compatible with the hospital. However, we also wanted feedback from both patients and therapists at the hospital throughout the design. It was decided to build a smaller, simpler robot, which could be taken to the hospital and tested promptly. Feedback could then be used to extend the simpler robot with additional DOF's in order to achieve more complex movements.

There were three options for a simpler robot configuration: 1 linear DOF, 1 rotary DOF, or 2-DOF. The primary motions we need to target include circular, horizontal linear (knee extensions), vertical linear (leg lifts), and more complex movements like gait trajectories. These also need to be adjustable to accommodate a variety of body sizes. A linear DOF could target knee extension or leg lifts, whereas a rotary DOF would only be able to target a weak approximation of a gait trajectory (*i.e.* a circular motion more similar to pedalling a bike). A 2 DOF system could accomplish more types of motions, for example doing both leg lifts and knee extensions, or more closely approximating a gait trajectory.

Ultimately the single rotary DOF was excluded because its movement was not relate to any of the bed-bound exercises. The 2-DOF was excluded because *(a)* it was determined to be too complex for the scope of this thesis, and *(b)* it was decided the final configuration of a multi-DOF device would be left to future iterations after receiving feedback from the hospital. Therefore, the final design is a single linear DOF.

The linear DOF could either be horizontal or vertical. ...

2.1.2 Linear Actuator Type

There are several ways of actuating a linear motion. This includes hydraulics & pneumatics, ballscrews, and belt drives.

Hydraulics & Pneumatics: Hydraulics and pneumatics use pressurized liquid or air to achieve linear actuation. They require more consideration than other off-the-shelf solutions, such as valves....

Table 2.1: Benefits of Actuator Types (x = benefit)

Type	**Cost**	**Ease of Setup**	**Backdrivable**	**Precision**
Ballscrew			x	x
Belt Drive	x	x	x	
Pneumatic/Hydraulic	x		x	

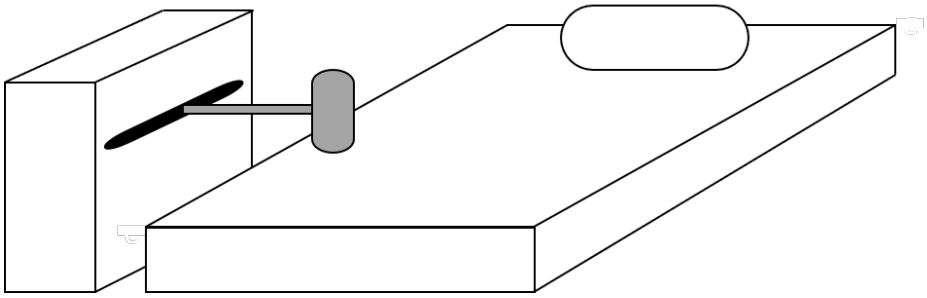

Figure 2.1: Off-bed Design

Ballscrew: Ball screws offer high precision and very low backlash. However, the rehabilitation needs to move quickly, and so the ballscrew would require a very large pitch, or the motor would need to powerful enough to supply high torque at high speed. Cost restraints therefore make using a ballscrew impractical.

Belt Drive: Belt drives are in general less expensive and can achieve high speeds. However, the belt introduces flexibility into the actuator which may affect the precision of the controller. However, precision in this application is not imperative since small errors will not be perceived by the patient. Furthermore, flexibility and belt slip at high torques could actually act as worst-case safety measures.

2.1.3 Placement Relative to Bed

The device needs to support the patient's leg and provide horizontal assistive forces through the leg flexion/extension exercise. Several possible structural configurations were identified. The device could sit beside the bed, with the foot plate extending horizontally to support the patient's foot (see Fig 2.1). The device could sit on the bed, under the patient's leg (see

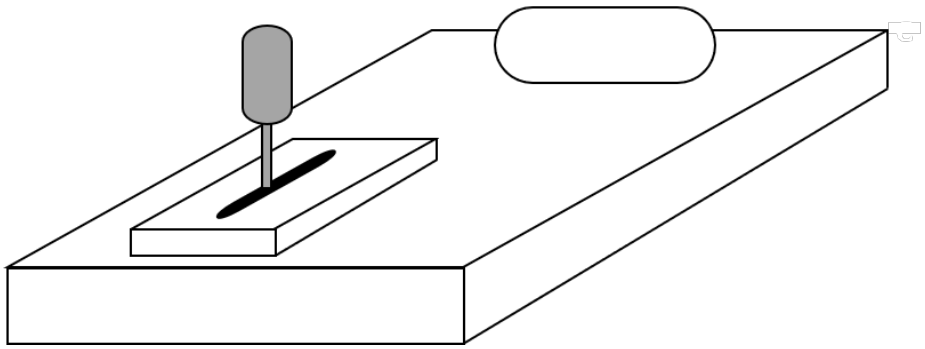

Figure 2.2: On-bed Design

Table 2.2: Structural Configuration

	On-Bed	**Beside-Bed**
Pros	Takes less space in room	Easy to set up
	Better support of patient's leg	Easier to extend DOF's
Cons	Harder to set up	Requires room beside bed
	May obstruct patient	Large moment on footplate
		Must move patient to edge of bed

Fig 2.2), similar to a passive range of motion machine. Each configurations has pros and cons, as listed in Table. 2.1.3.

The deciding factors were the moment applied by the leg on the footplate, and the space in the room. Two forces are created by the user: the applied force pushing or pulling the footplate (F_a), and the weight of the leg on the heel support (F_w). The on-bed design sits under the leg, so the moment created by the weight of the leg and its moment arm (R_x) can be minimized by minimizing the R_x. If the device sits beside the bed, the footplate must extend horizontally outwards, resulting in the weight of the leg creating a moment on the actuator throught that distance (R), which is harder to minimize (Fig. 2.4). The length of the footplate extension can be shortened to reduce the moment, however this would require

that the patient is transferred closer to the edge of the bed, which is both time consuming and possibly hazardous. Additionally, the device would require room beside the bed. After observing several hospital rooms in the stroke ward, it was clear that this space is not always available, due to equipment, furniture, and crowded rooms which can contain up to four beds. The on-bed design sits completely on the bed and so should be compatible with any room. The moment applied to the footplate due to the weight of the leg is smaller as it supports the leg from below. Finally, it can be positioned on the bed in a way that suits the patient, instead of having to move the patient to the edge.

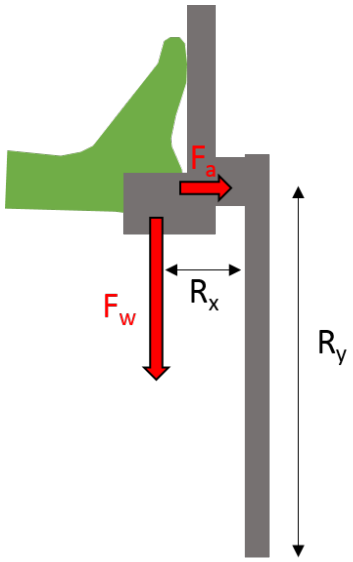

Figure 2.3: Forces and moment arms for on-bed design

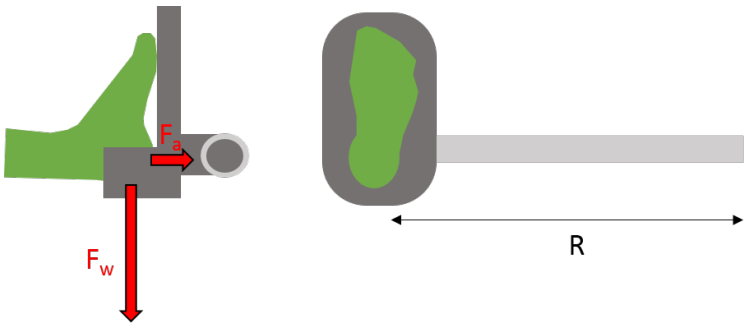

Figure 2.4: Forces and moment arm for off-bed design

2.1.4 Computer v.s. Microcontroller

The controller can be executed either from a microcontroller or from a full computer. The microcoontroller is cheaper, easier to set up for real time code execution, and has built in I/O's. Computers have more processing power, memory, can easily be networked or connected wirelessly, but require external data acquisition. Special care must be used to set up a real time system which can execute control loops with minimized latencies.

For a 1-DOF device, a microcontroller is the typical choice given that the control is simple and does not require much processing power or memory. However, eventually extra DOF's may be added to our system to the point where a microcontroller is not suitable. The computer could also be used to run the user interface (UI). Therefore we chose to use a computer to run the control software.

2.1.5 Secondary Leg Support

When therapists assist patients with the knee extension exercise they support the patients leg at two points: at the heel and under the knee. From the heel they hold the leg and apply force to assist or resist the motion. The hand under the knee also supports the leg, and keeps the leg from rotating out of plane.

2.2 Overview

The final design consists of one linear degree of freedom, actuated by a belt drive, which sits on the bed under the patient's leg. The footplate has heel support and a Velcro strap to keep the foot in place, and is height adjustable to accommodate a variety of leg sizes. There is also an adjustable end stop to limit the length of the stroke, which can be customized to each individual in order to prevent leg over-extension. Figure 2.5 shows a CAD rendering of the design with the frame, footplate, and actuator. Figure ... shows the device as of July 2019, with the adjustable end stop.

Figure 2.5: The rehabilitation device

2.3 Structural Design

2.3.1 Frame

The frame holds the actuator, footplate, and electronics (Fig. 2.6). It should fit on a bed, under one of the patient's leg, without obstructing the other leg or the patient's body. The frame is constructed out of 1x1' aluminium profile, as shown in Figure REF. The frame can be clamped or strapped to the bed. Future designs could include custom mounts to specific beds by attaching to the footboard or sides. The frame also contains an adjustable physical end stop which is used to control the maximum stroke length of the actuator. This can be adjusted to prevent over-extension of the leg.

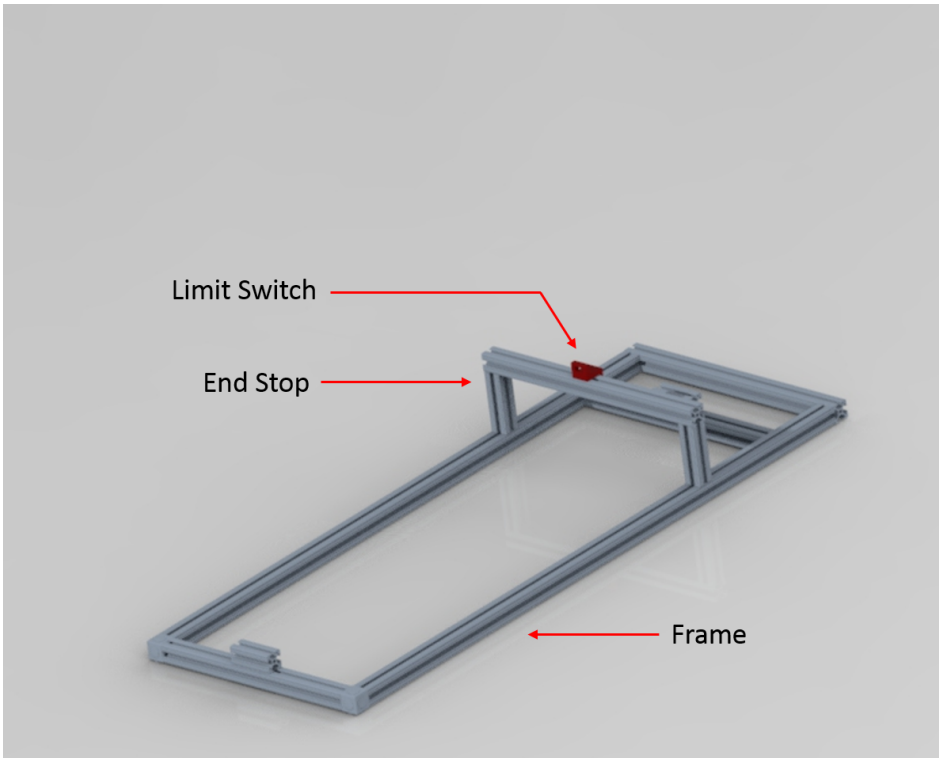

Figure 2.6: The structure of the frame

2.3.2 Footplate

The footplate attaches to the belt drive and should be height adjustable. The footplate both supports the leg and is a the interface through which the robot and user interact. It is shown in Fig 2.7.

An aluminium profile extends from the base on the belt drive. An off-the-shelf carriage with a handle lock is used to adjust the height. Mounting plates are used to attach a load cell to the carriage, which is placed near the bottom of the footplate. The footplate is cut from aluminium 1/8" plate.

The load cell measures force applied horizontally; however it may also register force due to moments placed on the footplate either from the weight of the leg or from applying forces near the top of the footplate. To minimize these erroneous measures, support brackets were created to transfer vertical forces from the footplate directly to the adjustable base, thereby reducing the effect of the moment. These brackets are shown in fig REF, and were 3D printed

Figure 2.7: The structure of the frame

with PLA.

To hold the foot in place, a custom heel support was created (Fig REF). It contains housing for an infra-red sensor which can detect whether the foot is in place for safety. The heel support was also 3D printed, using ABS.

2.4 Hardware

2.4.1 Actuator

The system is actuated using a Brushed DC Motor, gear reduction, coupling, and an off-the-shelf linear belt drive (Fig. 2.11). The Brushed DC Motor is a Maxon RE 50 200W Graphite Brush model (part number 370354). The gear reduction is the Maxon GP 52C Planetary Gearhead (part number 223090). The coupling is an Aluminium Flexible Shaft Coupling with a polyurethance spider from McMaster-Carr. The linear belt drive is a LB250-24 Belt Drive from Anaheim Automation. The motor is mounted to the belt drive using custom machined aluminium mounts.

The motor is required to move the actuator fast enough while also be able to apply adequate levels of force fro assistance or resistance. This motor and gear assembly was used

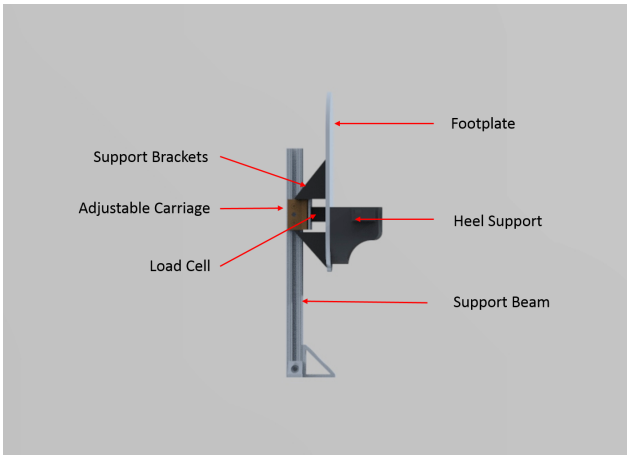

Figure 2.8: The structure of the footplate

Table 2.3: DC Motor Properties

Property	**Value**
Nominal Voltage	24V
Torque Constant	38.5 mNm/A
Speed Constant	248 rpm/V
Stall Torque	8.92 Nm
Nominal Speed	5680 rpm
Nominal Torque	405 mNm

from a previous project, and was found capable of fast of linear motion and applied force.

The maximum linear speed is in this case limited by the maximum input RPM to the gearhead, 6000RPM:

$$\begin{aligned} V_{max} &= \frac{p\omega_{max}}{GR}\frac{1min}{60s} \\ &= \frac{(0.122m)(6000RPM)}{(53)}\frac{1min}{60s} \\ &= 0.230m/s \end{aligned}$$

The nominal torque and speed are used to estimate the nominal linear speed and its

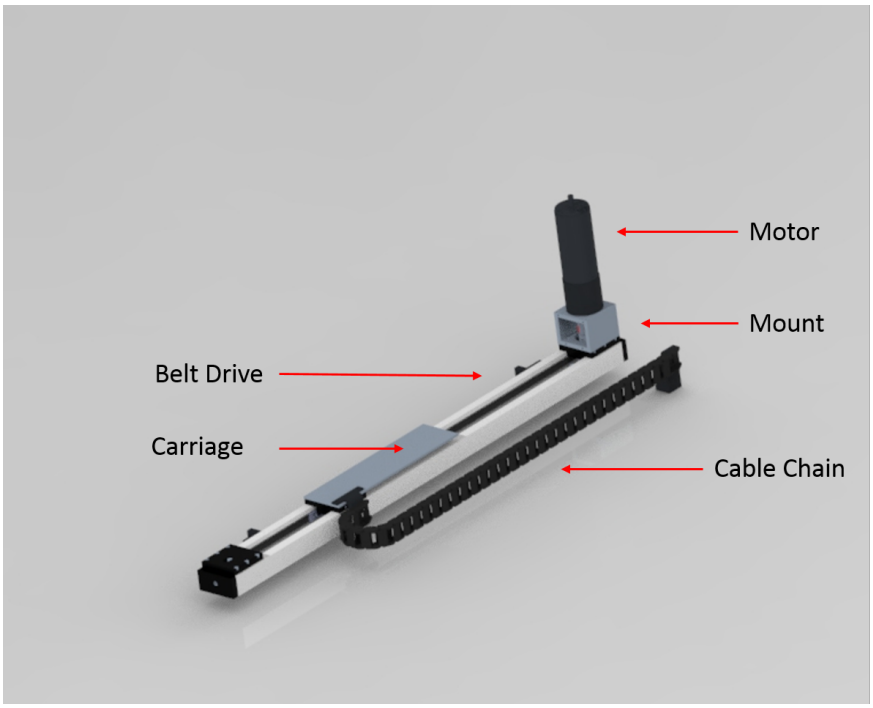

Figure 2.9: The belt drive and motor

Table 2.4: Gear Reduction Properties

Property	**Value**
Reduction	53:1
Max Speed Input	6000 rpm
Max Torque Input	45 Nm

associated nominal applied force:

$$V_{nom} = \frac{p\omega_{nom}}{GR}\frac{1min}{60s}$$
$$= 0.217m/s$$

Table 2.5: Belt Drive Properties

Property	**Value**
Stroke Length	24 in
Travel per Revolutions	4.8 in/rev
Maximum Travel Speed	100 in/s
Static Load capacity	1000 lb
Dynamic Load Capacity	200 lb
Pulley Radius	35 mm

$$\begin{aligned} F_{nom} &= \frac{\tau_{nom} GR}{R_{pulley}} \\ &= \frac{(0.405Nm)(53)}{0.0175} \\ &= 1227N = 275lbs \end{aligned}$$

The maximum speed of 230 mm/s is adequate for our purposes, as the system is designed for stroke rehabilitation which is a slow moving process. The nominal force of 275lbs is much higher than the assistive forces seen during traditional therapy (which therapists estimate to be up to 20 lbs). In the future, a lower gear reduction could be used to increase the maximum speed to accommodate more advanced patients.

Table 2.6: System Maximums

Property	**Value**
Maximum Linear Speed	230 mm/s
Nominal Linear Speed	218 mm/s
Nominal Force	275 lbs

2.4.2 Sensors

A total of five sensors are used on the device, including: a load cell, two limit switches, a quadrature encoder, and an infrared (IR) proximity sensor. The load cell and encoder are

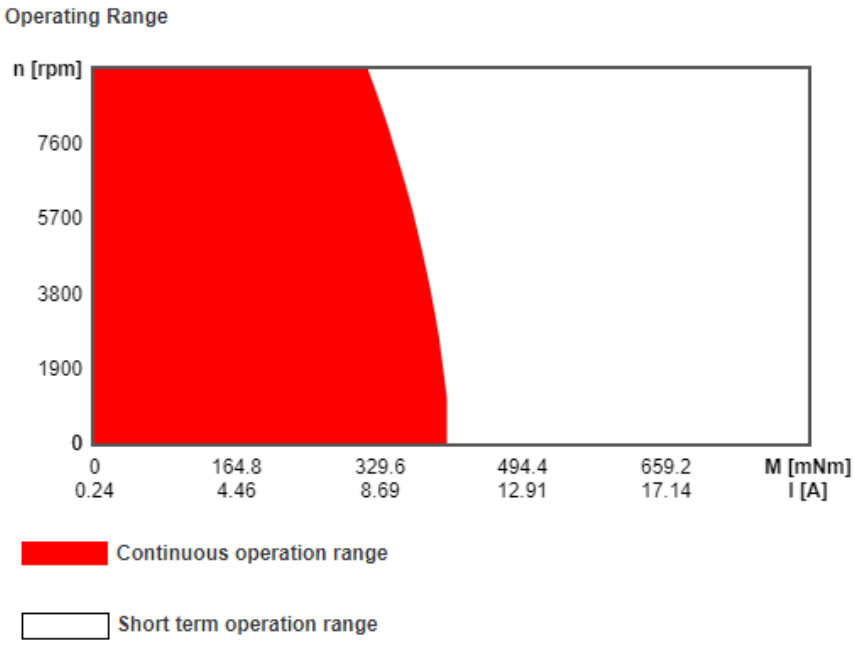

Figure 2.10: Operating Range of the Motor
www.maxongroup.com/maxon/view/product/motor/dcmotor/re/re50/370354

used to measure force and relative change in position for use in the controller; the limit switches are used to sense if the carriage as reached the end of the stroke in the front or back of the device, and the IR sensor detects the user's foot on the footplate to ensure the device can be deactivated if the foot is removed or if no one is using it.

2.5 Electronics and Power

Custom electronics were needed in order to power and operate the sensors, and send command signals to the motor. At first this was achieved using a protoboard (see Fig.). Wiring and space restrictions became a problem and motivated the a redesign using a printed circuit board (PCB), along with a 3D printed case to enclose the electronics. The PCB has four functions: supply power to the sensors, route signals to and from the DAQ, process or filter signals as necessary, and implement basic safety precautions.

- **Power Distribution**: The entire device should run off of a single power supply so that only one wall outlet needs to be used. A 48V supply was chosen, since voltage is required by the motor. The sensors require 24V (Load Cell), and 5V (the IR sensor), and 10V is required to send commands to the motor controller. Therefore the PCB has three step-down voltage regulators at these three levels.

- **Signal Routing**: To streamline the wiring, the PCB takes in three main wires: a DB9 serial connection from the sensors on the footplate, a DB9 serial connection from the power supply and motor controller, and a DB37 serial connection from the DAQ. These cables are connected to the PCB case and routed to the correct locations.

- **Signal Processing**: There are a few processing steps taken on the PCB. An adjustable first order low pass filter is used to clean noise from the force measurements. The motor commands from the DAQ are also processed. The DAQ can output 0-5V from the analog pin, however the motor controller acceptc commands from -10 to 10V. The DAQ commands are amplified by 4 and shifted down by 10V using an Op-Amp, such that the 0-5V output can be mapped to the +/- 10V input.

- **Safety**: The PCB implements four safety checks. First, the motor commands is checked to ensure it is within an adjustable range. This range can be changed using potentiometers. A Watchdog timer is also used to ensure the software has not stalled. Every timestep, the software send a pulse to the timer – if a pulse is missed within a certain delay, the Watchdog timer is triggered. There is also an emergency stop button which can be pressed at any time by operator or user. Th E-stop signal, Watchdog timer and motor command limits are combined in a logical AND gate, and the resulting output is sent to the enable pin on the motor controller. Any trigger from the three safety signals will disable the motor and stop the device.

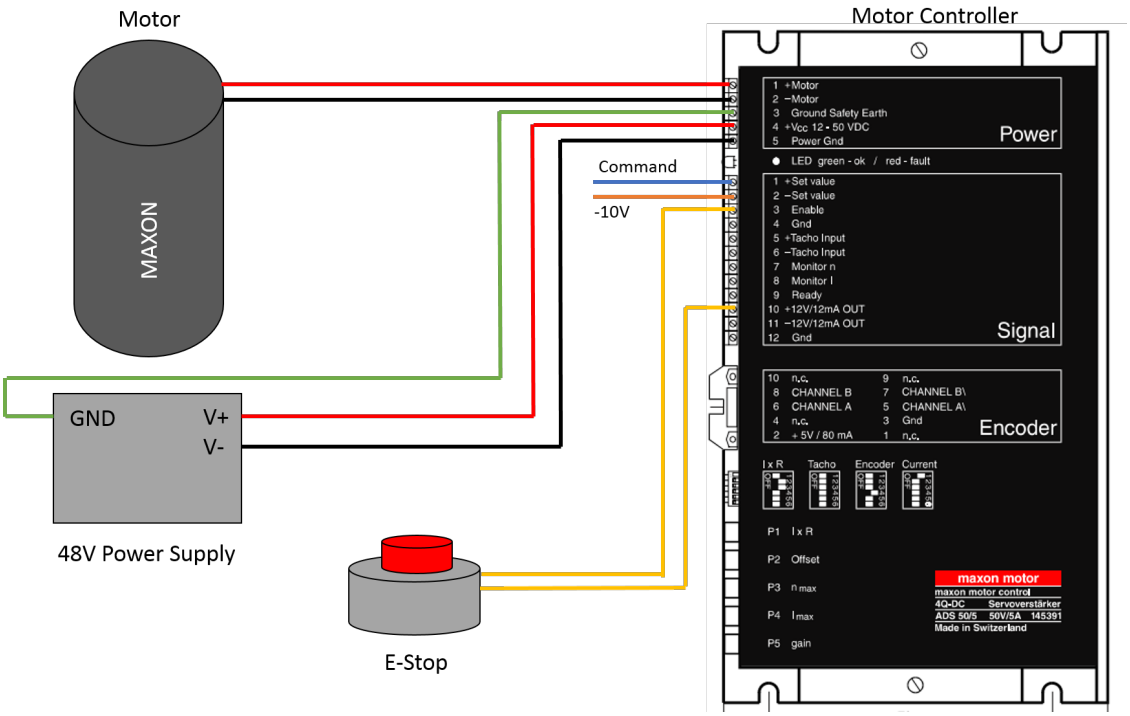

Figure 2.11: The belt drive and motor

2.5.1 Components

2.5.2 PCB Layout

2.6 Data Acquisition

Data acquisition is handled by a LabJack T7 DAQ. This model has 14 analog inputs, 2 analog outputs, 23 digital I/O's, and is also capable of handling quadrature encoder signals and high-speed data streaming. The device can interface with the computer through a Ethernet, USB, or wifi (Ethernet was chosen for this project as it is the fastest and most reliable). The front-end Modbus TCP interface is conveniently packaged in the C/C++ LJM library, discussed further in chapter ...

Table 2.7: PCB Components

Component	**Description**
24 V DC/DC Converter	Converts 48V to 24V
10 & 5V Linear Regulator	Converts 48V to 10V and 5V
OP-AMP	Signal Processing
Comparator	Checks to ensure commands are not above/below threshold
4 Channel AND Gate	Only enables operation if all saftey checks read ok
Watchdog Timer	Ensures controller is still running

2.7 Computer

The control software and UI are run from an Intel NUC computer (Fig REF). It has the following specifications:

Table 2.8: PCB Components

Processor	2.2 GHz Intel Core i5
RAM	8GB DDR4
Memory	250GB Solid State

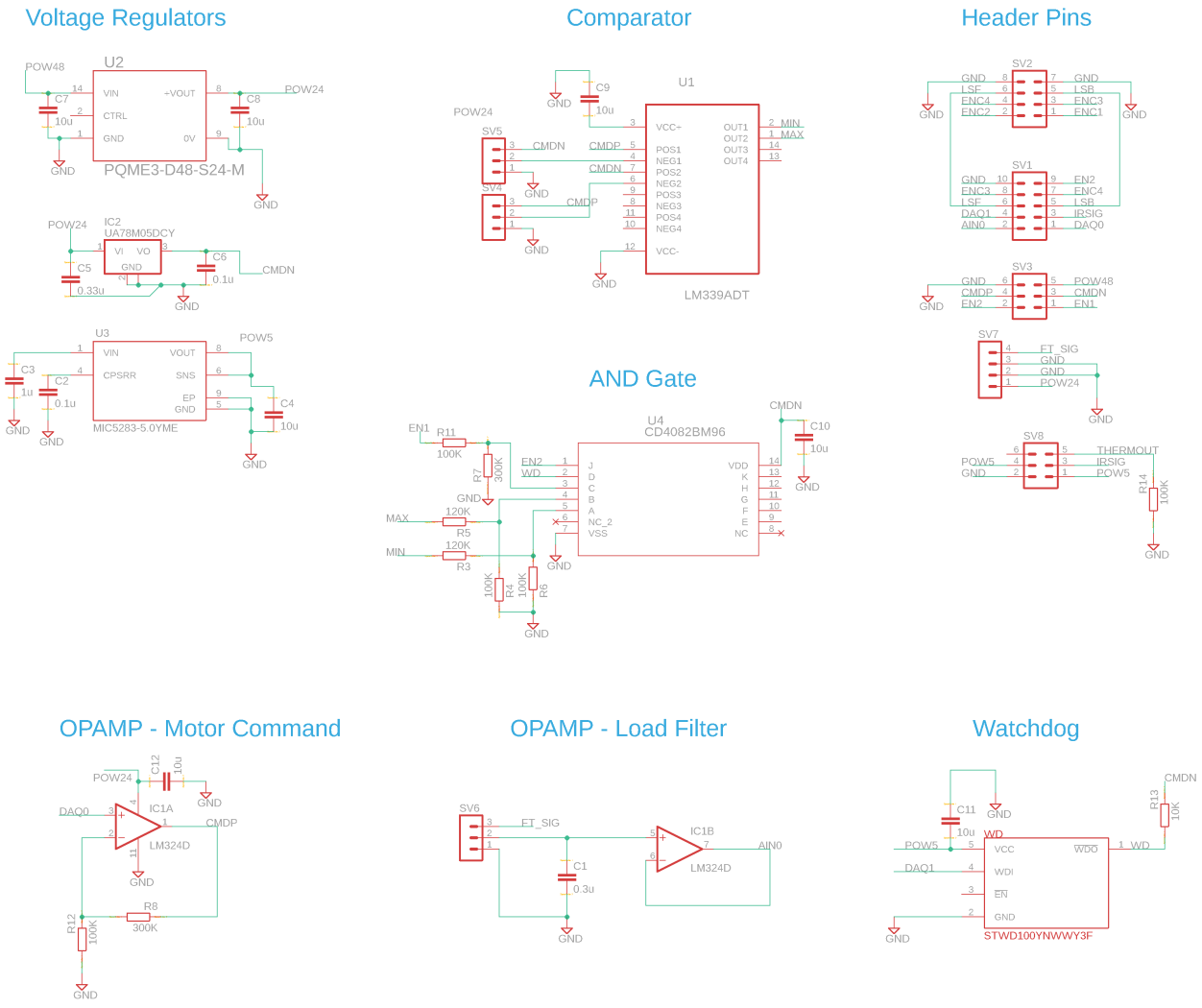

Figure 2.12: Printed Circuit Board

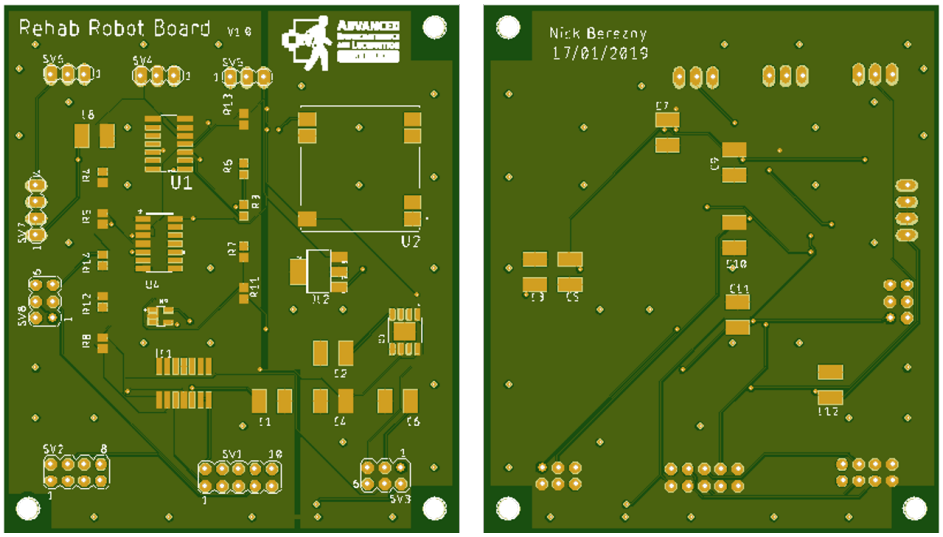

Figure 2.13: The PCB Layout

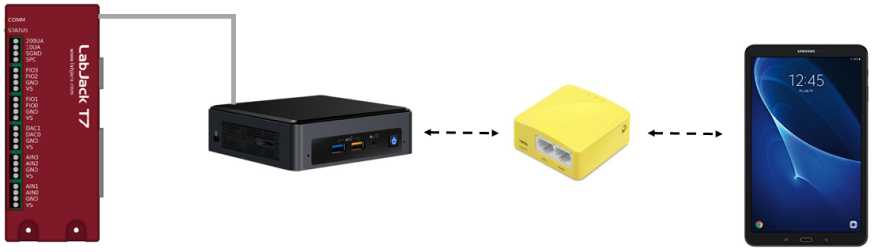

Figure 2.14: The Computer System, with DAQ, Router, and Tablet

Chapter 3

Controller and Software

3.1 Overview

The controller and software must allow the device to perform the required tasks as laid out in chapter 2. This includes providing assistance to the patient through the exercise, but not to the extent that the patient can become passive and allow the robot to drive the motion. The level of assistance should be adjustable so that the operator can tailor the session to the capabilities of the patient. The software must also engage the patient using virtual environments and games. These must be related to the motion of the exercise, such that the robot becomes a control input into the game, and the patient is rewarded for following the desired trajectory. Immersion within the virtual environment could also be bolstered by haptic feedback, or the physical rendering of virtual forces to the patient through the robotic device.

3.2 Impedance & Admittance Controller

Both position and force control may behave unpredictably when interacting with an unknown environment, and may not be well suited for the purposes of assisting patients through rehabilitation movements. Position control would not provide assistance as needed, but

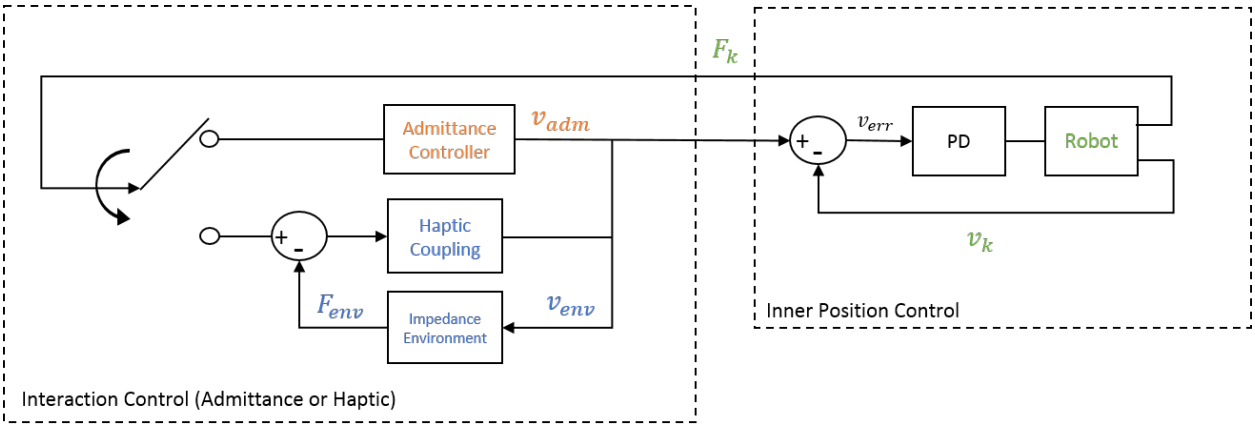

Figure 3.1: Virtual Haptic Coupling for Admittance Control

would simply force the user's leg through the motion, similar to a passive motion machine. Force control is not sufficient since it does not allow the controller to follow a predefined trajectory. Interaction control, also known as impedance/admittance control, was introduced in [33]. Instead of controlling position or force directly, interaction control seeks to control the dynamic relation between the position and force variables.

There are two possible causalities – an impedance, which accepts a velocity/position and renders a force, and an admittance, which accepts force and renders a velocity/position. An interaction controller will chose one of these causalities, and will sense input from the physical device, calculate the output variables based on some virtual desired system, and then command the device to that output. A commonly used virtual system is the mass-spring-damper system shown in Fig. 3.2.

Impedance control measures the position and velocity of the robot, and then calculates the force generated by the virtual system, as in Eqn. 3.13 & 3.2. The device must then use force control to achieve the desired output force, which is usually done by computed-torque methods.

$$F = ZV_R \tag{3.1}$$

$$F = m\ddot{x} + b\dot{x} + k(x - x_{des}) \tag{3.2}$$

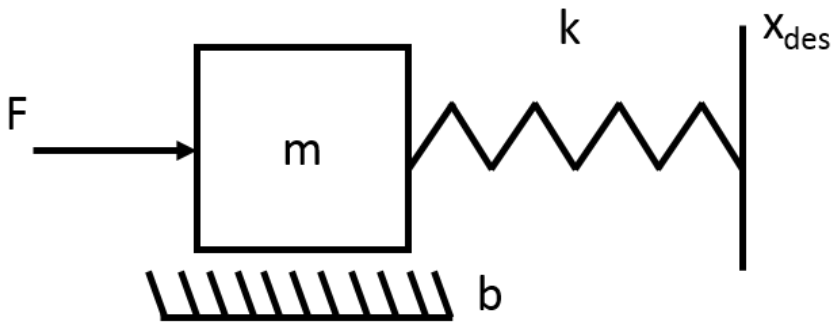

Figure 3.2: A mass-spring-damper system, with the origin of the spring set to some desired trajectory

Using this control law, the end effector of the robot will behave like the system above, about point x_{des}, when displaced by the external environment. If, for example, an operator physical moved the robot, it would respond as if it were the mass-spring-damper system, *i.e.* it would apply a resistive force related to the extent to which it was displaced. This control law rests on a significant assumption – that the robot itself has no inherent impedance between the actuator and the the end-effector. It is possible to cancel some of the impedance of the robot using ..., however this is less common and generally takes more time and is less successful (ref). This is an idealization - there will also be some level of friction, mass, or other impeding factors in the motor, gears, links, etc. Therefore, impedance control should only be considered for robots made of low impedance components, such as belt drives and direct-drives.

Admittance control is essentially the inverse of impedance control, in that it accepts an external force and then renders a corresponding displacement. Fundamentally, admittance control can be described by the following equation, where F_R is the force applied to the robot, Y is the desired admittance, and V is the 'flow' variable vectors (e.g. position and velocity) :

$$V = Y F_R \tag{3.3}$$

Once again, it is common to define admittance in terms of a mass-spring-damper system (3.4). Here, x_a and v_a are the "simulated" position and velocity determined by the admittance controller. Additionally, to actually render the desired position and velocity, an outer-loop PD position controller is used (3.5), with x_a and v_a acting as the desired position and velocity.

$$\begin{bmatrix} \dot{x}_a \\ \dot{v}_a \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{k}{m} & \frac{b}{m} \end{bmatrix} \begin{bmatrix} x_a \\ v_a \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F_R \tag{3.4}$$

$$F = P(x - x_a) + D(v - v_a) \tag{3.5}$$

Admittance control assumes that the robot has infinite impedance, or that all hardware between the actuator and the end-effector is infinitely stiff, infinitely damped, etc. This, once again, is an idealization – and therefore admittance control should only be considered when the robot has high impedance, i.e. when using large gear reductions and rigid links.

Another point of consideration when choosing between impedance and admittance control is its stability when encountering different types of environments. Impedance control works better when interacting with an admittance (rigid environments), whereas admittance control works better when interacting with an impedance (weak environment / free space). This project targets acute stroke patients who typically do not have much leg strength and rely heavily on assistance from the therapist during therapy. Therefore, these patients can be modeled as an impedance. Furthermore, the motor used on the robot has a high gear reduction which contains substantial impedance. Both of these favour using admittance control.

Admittance control is implemented by taking the Z-transform of Eqn. 3.4, updating the desired position based on these discrete step equations, and then using PD controller to move to this desired position. ...

The linear admittance control system in Eqn. 3.6 is discretized according to the Eqn. 3.7 & 3.8.

$$\dot{X} = AX + BF \tag{3.6}$$

$$A_d = e^{At} \tag{3.7}$$

$$B_d = A^{-1}(A_d - I)B \tag{3.8}$$

3.2.1 Velocity Profile

The final step is to set the desired position a suitable trajectory. The system is 1-DOF, so this can be done simply by creating a velocity profile. The velocity profile is defined in terms of the total length of the linear stroke (x_{max}) as well as the maximum velocity that we want to have at any point (v_{max}). At first, a quadratic relation was used, creating a parabola with zero velocity at each endpoint and v_{max} in the center. This was found to feel unnatural. The final profile is shown in Fig. 3.3, with constant acceleration to v_{max}, a plateau of constant velocity, followed by a constant deceleration. The same profile is inverted to move back at the end of the stroke. The transition points were set empirically, with acceleration up to $0.4x_{max}$, and deceleration beginning at $0.6x_{max}$.

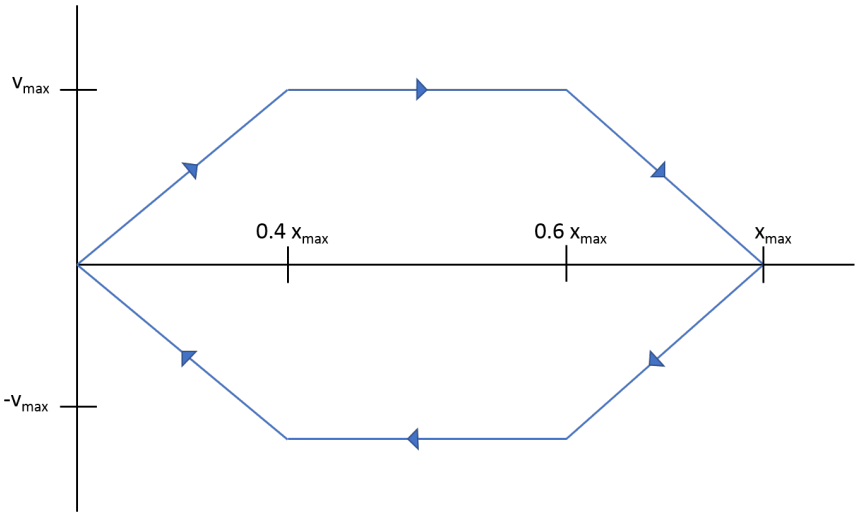

Figure 3.3: Velocity Profile of the Desired Trajectory

3.2.2 Operational Modes Using Admittance Control

Two operational modes use the admittance control with the mass-spring-damper system – trajectory tracking with assistance, and trajectory tracking with resistance.

Trajectory Tracking with Assistance: The mass-spring-damper system is used with a moving spring origin set to the trajectory as defined by the velocity profile. The spring stiffness represents the level of assistance, with stiffer springs providing more assistive force. The assistance is low when the error is low, therefore only providing assistance as needed.

Trajectory Tracking with Resistance: The spring term is omitted (*i.e.* k = 0.0) so that the user only feels damping and mass. The damper term now represents viscous resistance, with higher damping parameters creating greater resistance. Without the spring term, there are no assistive forces and so the patient must provide all motion on their own. This mode could be used for more capable patient's who need a challenge, or for strength building.

3.3 Haptic Coupling

Defining the desired impedance or admittance in terms of a mass-spring-damper system is not always convenient. Ideally, one could define an arbitrary impedance or admittance function relating F and V, giving complete control of the behaviour of the robot-environment interaction, while also ensuring stability. The dual goals of imposing a

A method for achieving arbitrary interaction and remaining stable is discussed in [39], using what is called a "haptic coupling". This effectively decouples the design on the admittance/impedance and analysis of the stability of the robot. Consider again the two-port model of the robot-environment interaction. The haptic coupling is a virtual passive system placed in between the two, which acts as a low pass filter on the variables. This limits the range of renderable impedances, and while it does alter the desired impedance to some extent, the trade off is guaranteed stability. Using the concept of the haptic coupling, a simple physics engine was designed which works in conjunction with the games and visualization.

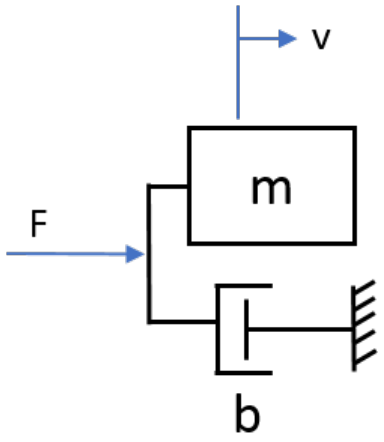

Figure 3.4: Virtual Haptic Coupling for Admittance Control

In addition to decoupling stability and the design of the haptic environment, the haptic coupling also allows both impedance and admittance to be used together. For example, a robot may use admittance control due to large amounts of drive friction, but the haptic environment may need to be defined as an impedance, *i.e.* using position to calculate the desired haptic force.

$$Z(s) = \frac{F}{v} = ms + b \tag{3.9}$$

$$Z(z) = \frac{mb(z-1)}{(m+bT)z - m} \tag{3.10}$$

The z-transform yields the control step algorithm in Eqn. 3.11. These are used to calculate the present time environmental position/velocity.

$$v_k = \frac{m}{T} v_{k-1} + \frac{f_{k-1} - f}{\frac{m}{T} + b} \tag{3.11}$$

$$x_k = x_{k-1} + v_k T \tag{3.12}$$

The underlying virtual physics should have an impedance causality, and so the environmental force is calculated based on this position/velocity pair. This mapping can be an arbitrary function, thanks to the haptic coupling (Eqn. 3.13).

$$f_k = \mathcal{F}(x_k, v_k) \tag{3.13}$$

For example, to achieve the standard admittance control from the previous section, the impedance mapping can generate the spring component (with the damping and mass contributed by the haptic coupling), as in Eqn. 3.14

$$f_k = K(x_k - x_{des}) \tag{3.14}$$

With the haptic coupling, more complex environments can be created. For example, contact between the user and a virtual object can be rendered. The interaction force is set to zero when there is no contact, allowing for free motion. When in contact, the interaction force can be modelled a number of ways, including with a simple spring

$$f_k = \begin{cases} \text{no contact} & f_k = 0 \\ \text{contact} & f_k = k(x_{user} - x_{dist}) \end{cases}$$

3.3.1 Operational Modes Using the Haptic Coupling

Only a single mode is implemented using the haptic coupling – a simple interactive balance game. The player is tasked with maintaining a static position within a small margin of error. Disturbances are launched at the player from both directions, which contact the user and produce a haptic force

3.3.2 Physics Engine

There physics involves the movement of two objects, the character and the disturbance, as well as the interaction force between them. There are three cases:

1. Disturbance is out of play: The disturbance has left the environment. A new one is randomly spawned to either the right or left of the environment, and given a fixed initial velocity.

2. No Contact: There is no contact. The position of each object is updated, with the character's position set equal to the position of the device, and the disturbances position set according to its constant velocity.

3. Contact: The character's sphere overlaps the disturbances sphere, and are therefore in contact. The contact force is modelled a linear spring, with the distance between the objects' centres times the contact stiffness equal to the interaction force.

3.4 Tuning Control Gains

Some control gains must tuned for optimal performance and stability, which include:

1. P & D for the position PD controller

2. b & m for the haptic coupling ()

3. I_{max} on the motor controller, giving the current command given to the motor at the maximum motor command of $\pm 10V$

Other controller gains are set based on the desired behaviour of the device – these require bounds so that the user does not enter bad parameters:

1. K,B,M for the admittance controller

2. V_{max} for trajectory generation

3. F_{max} for the maximum force generated by the system

...

First, the PD controller can be tuned separately from the admittance. The PD controller can be though of as another impedance existing within the system, and should therefore be tuned as high (stiff) as possible for optimal performance of the admittance controller. The P gain should be set such that the largest position error corresponds with the maximum motor command.

The maximum motor command at the motor controller is $\pm 10V$, however the motor command sent from the DAQ is multiplied by a factor of approximately 4 and referenced to a constant $10V$. Therefore the maximum command sent from the controller is between 0 (maximum current in one direction) to 5 (maximum current in the other direction).

The maximum displacement can be calculated by using the maximum velocity achievable by the actuator. In this case that is $9in/s$, which corresponds to a maximum displacement of about $0.23mm/step$ for a controller running at 1000 Hz. However, designing for this displacement will be conservative in that the gains will be very small. Furthermore, since the system is interacting with a person whose behaviour is unknown, there is always the chance the system will be driven beyond the limits. Therefore it was decided to design the PD controller based on a maximum step of 2mm, and to then try and enforce the maximum velocity of $9in/s$ through visual feedback to the user.

Given these parameters, the P gain should be set to 2.4:

$$cmd_{max} = P\Delta x$$
$$P = \frac{cmd_{max}}{\Delta x} = \frac{4.8}{2} = 2.4$$

Next, the D gain is determined empirically by increasing it until there is no overshoot to a step response ....

The limits on the admittance parameters can now be determined. The admittance system can be approximated by the following:

$$\Delta v = -\frac{K}{M}x - \frac{B}{M} + \frac{1}{M}F \tag{3.16}$$

We want to avoid producing a change in position greater than 2mm as per the PD controller tuning. Therefore, the maximum velocity of the admittance model should not exceed 2000 mm/s (for a controller running at 1 kHz). Therefore, Δv should be equal to zero at this maximum velocity and at the maximum achievable spring force, *i.e.* when the error between actual position and reference position is maximum. Therefore:

$$0 = -K(400mm) - B(2000mm/s) + 222.4N \tag{3.17}$$

$$K \leq 0.556 - 5B \tag{3.18}$$

he mass is factored out, and the maximum error in position, maximum velocity, and the maximum force are entered. This defines space bounded by a line in which the (K, B) vector must lie. Furthermore, there is a minimum constraint on the damping (B) since it is the dissipative element and is therefore required for passivity and stability. This constraint, B_{min} is found empirically by setting K high and finding the minimum B to make the system behave in a stable manner. ...

The mass does not affect the rendered impedance, but instead determines the systems sensitivity to force. The smaller the mass, the smaller the force that is required to achieve a certain state ... There is therefore no real maximum mass, however at some point a high mass will make the system impossible to move. There is a minimum mass, below which the sensitivity to force allows sensor noise or unmodeled dynamics to effect the system to an undesirable extent.

...

Finally, the haptic coupling parameters m and b should be set. As was explained in Sec. , these parameters give the controller some minimum impedance so that the haptics do not try to render an impedance too close to free motion. Therefore, the parameters are set to the empirically derived B_{min} and its pair K_{min}. This decouples the haptic design from the stability of the controller by imposing the minimum impedance for the stability. The maximum constraints are however still not imposed. Fortunately, the haptics explicitly calculate either the desired force or the desired next position, and therefore the constraints

on these parameters can be directly imposed.

3.5 Software

The controller needs to be implemented in software, along with communication, data logging, and a User Interface for setting up and displaying visuals. The software is split into two parts: the control software and the user interface software. The entire codebase is available on GitHub at https://github.com/nickberezny/RehabRobot. It is built to run on real time enabled Linux, and should run on any hardware (assuming it has sufficient memory and RAM, and an ethernet port to connect with the DAQ). This modularity should make it easy to use this code in future iterations of the project which may use different hardware. To date, it has been tested on the Intel NUC used in this project, as well as on a Raspberry Pi 3B.

3.5.1 Real Time Operation and Linux

A real time system is one in which the timing of the computations significantly affects the results [40] – in other words, a missed computational deadline can be detrimental to the performance of the system. Systems can either be soft real time or hard real time, a distinction made based on the danger associated with missing a deadline. Hard real time system will cause serious harm or damage if a deadline is missed (*e.g.* the control systems on a Jet), while the consequences are typically minor in soft real time systems. The distinction can be vague, but in general hard real time systems require programs which are mathematically proven to be real time (REF), whereas soft real time systems can get away with less stringent requirements.

The consequences of a missed deadline in our system is at worst sending the wrong motor torque command to the actuator, which in the absolute worst case scenario will cause the maximum torque command to be issued. However, motor torque commands are limited for safety (see sec...), and the operator and user have access to an emergency stop. Therefore,

we deemed that the system requires only soft real time capabilities.

Linux is an operating system (OS) based on UNIX which is open source and comes in variety of versions or flavours, including Ubuntu. Linux can be used as a standard home computing operating system like Windows or Mac, and is also found commonly in embedded system where the OS is highly tailored to a specific set of hardware. (refs!). The Linux kernel is not inherently real time – there is a patch available which adds soft real time capabilities to the OS. This is known as Real-time Linux or RTLinux, and is created using the Preempt RT patch [41]. The patch allows programmings to use the following real time functionalities, as defined in the real time standard extension of the original Portable Operating System Interface for Computing Environments (POSIX) [42].

- **Threads**: POSIX threads, or pthreads, are used for parallel computing. Multiple threads can run under a single process and share its memory space.
- **Priority Scheduling**: Allows setting thread priorities, with higher priority processes pre-empting lower priority processes. This helps critical control code to run uninterrupted by less important work.
- **Mutexes**: A Mutual Exclusions Object is used to orchestrate data sharing between threads. Once one thread takes ownership of a Mutex, others cannot access it before it is unlocked. This prevent other threads from changing the data before the previous thread is finished.
- **Memory Locking**: Locks memory into RAM to prevent the program from accessing the hard disk (Page Faults), which can cause delays.

3.5.2 Controller

The controller is written in C, and is designed to achieve low latency so that it can meet sampling rate deadlines. The target frequency is 1kHz, or 1ms per time step. The controller on average runs at 1.05 ms per time step, or approximately 9.5 kHz. More information of delays and jitter in the control software can be found in Sec REF...

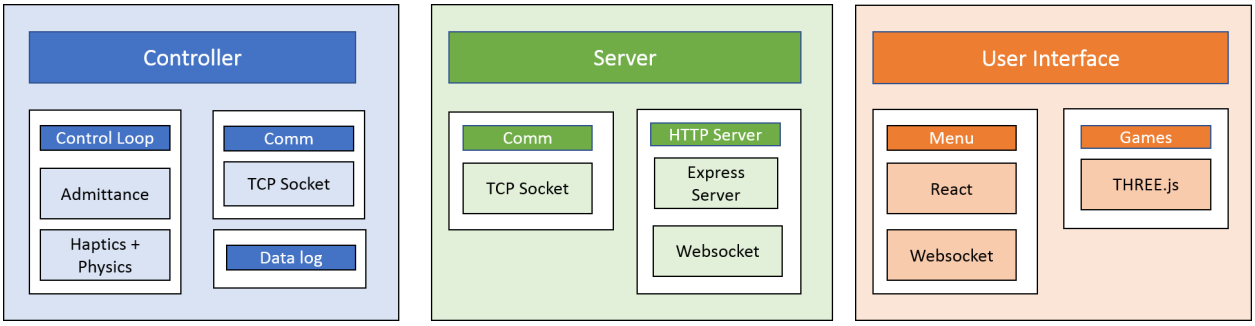

Figure 3.5: The Three Threads in the Control Software

The controller process is split into three threads: the controller, the server communication, and data logging, with priority levels of 98, 97, and 96 respectively. The controller has highest priority and can pre-empt the others to keep it running up to the desired frequency.

Data sharing between the three threads is handled by mutexes and a 10 dimension array of data structures. Each thread iterates through the data structure array. The control thread, being the highest priority, locks the first element. The other threads cannot lock the mutex and so have to wait. The control loop runs through one step, setting any data from the DAQ or otherwise to the data structure. Once the time step is finished, it unlocks the first element and locks the second, and continues. The communication thread can now lock the first element, and send the required data to the UI, such as the position of the device. Once complete, it unlocks the first element and waits for the next to be unlocked by the controller. Finally, the data logging thread can lock the first element, and print the required variables to another row in the data log file. This process then continues through the ten elements, with the threads going back to the start once finished with the tenth element...

Setup

Once code execution is started, there is a number of setup and initialization steps before jumping into the control loop.

1. Global variables are created.

2. The TCP socket is initialized for communication with the UI

3. Connection with the DAQ is established. The I/O's are setup as necessary (*e.g.* two digital pins are set as quadrature counters for reading the motor encoder)
4. The data log text file is created, and the header row is printed.
5. A total of 10 mutexes are created to regulate data access to the control data.
6. Memory is locked - no new variables should be allocated after this point
7. The Server code is executed in automatically in another terminal so that the user can connect to the UI.
8. The controller waits for input from the UI. Once received, the message is parsed and the control parameters are set.
9. Admittance control matrices Ad and Bd are created according to the discretization formulas in REF.
10. Once the UI sends the home command, the homing procedure is executed, with the carriage moving from front to back, to ensure the device starts from a consistent position, and to measure the total stroke length as set by the adjustable endstop.
11. Once the UI sends the run command, the force sensor is calibrated by taking 10 measurements, checking for outliers, and then setting the force bias to the average.
12. The initialization is now complete, and the three threads are run.

Control Loop Thread

The control loop can be broken down into four steps, which are executed repeatedly until the program is stopped either by a set end time or by a stop command from the UI.

1. The current robot state is determined by reading sensor values from the DAQ.
2. The selected control algorithm is run to determine the command (either admittance control or haptics with physics).

3. The command is sent to the DAQ to be realized by the motor.

4. The total change in time since the beginning of the step is determined, and the controller sleeps for any remaining time so that the step lasts approximately 1ms.

Server Communication Thread

This thread sends data to the UI so that the games can be updated. To prevent overloading the UI, the data is only sent every 10 time steps (every 10ms). This data is sent through the TCP socket.

Data Logging Thread

This thread logs relevant data to a text file. It does this by printing a predefined set of variables at every time step.

3.5.3 User Interface

The User Interface (UI) is used both by the operator and the patient. It can be used to set up the parameters for a session, display the visual feedback to the patient, and view the session data and past session data for progress monitoring. The UI should be intuitive, easy to access, and portable across different hardware and computers (including tablets, mobile devices, etc.).

There are a number of libraries for creating UI's. For example, GTK is a toolkit for creating UI's on Linux, Mac or Windows [43]. However, we want the UI to work on mobile devices as well (iPADs and Android Tablets), which are more portable and may be easier to position so that the patient can access it comfortably. Ideally, a library that works on Linux, Max, Windows, Android, and iOS. One option is to use web technology, given that all of these OS's have internet browsers. We therefore decided to created and serve a website from the computer through a local network, that could be access by any device which has access to the same local network.

The UI uses the following libraries/tools:

- **Node.js**: A Javascript runtime used to build JS based applications
- **React**: A component-based JavaScript UI framework
- **Redux**: State management for React
- **Express**: Node.js framework, used to create the UI's HTTP server
- **Next**: Server-side rendering for faster page loading
- **Material-UI**: A css framework based on Google's Material Design
- **Socket.io**: Library for creating websocket used in communication with host computer
- **Three.js**: A javascript 3D graphic library

Ultimately, the UI is built on React, which is a javascript library for creating graphical user interfaces on the web. Components like buttons or inputs can be customized and re-used through the app. This allows for easy customization for future project iteration that will likely need new features.

Layout & Pages

The UI consists of a menu and three pages: the setup page, the game page, and the settings page. The UI layout includes a top bar with menu access, the menu (which can be hidden or expanded), and the content.

Setup Page: Seen in Fig. 3.6, the setup page allows the therapist or operator to set the parameters for a therapy session. The therapist can set the desired game, the length of the session, velocity adn force limits, assistance levels, etc. In developer mode, controller gains can be tuned directly. The set button sends this information to the control prcoess.

Games Page: The game page displays the games or visuals to the patient. It contains the home and run buttons which are used to set up the device before starting the session.

Settings Page: This page contains any UI settings that one may want to change. For now, this only allows the operator to switch between Developer Mode and User Mode, which change the level of detail in the setup page.

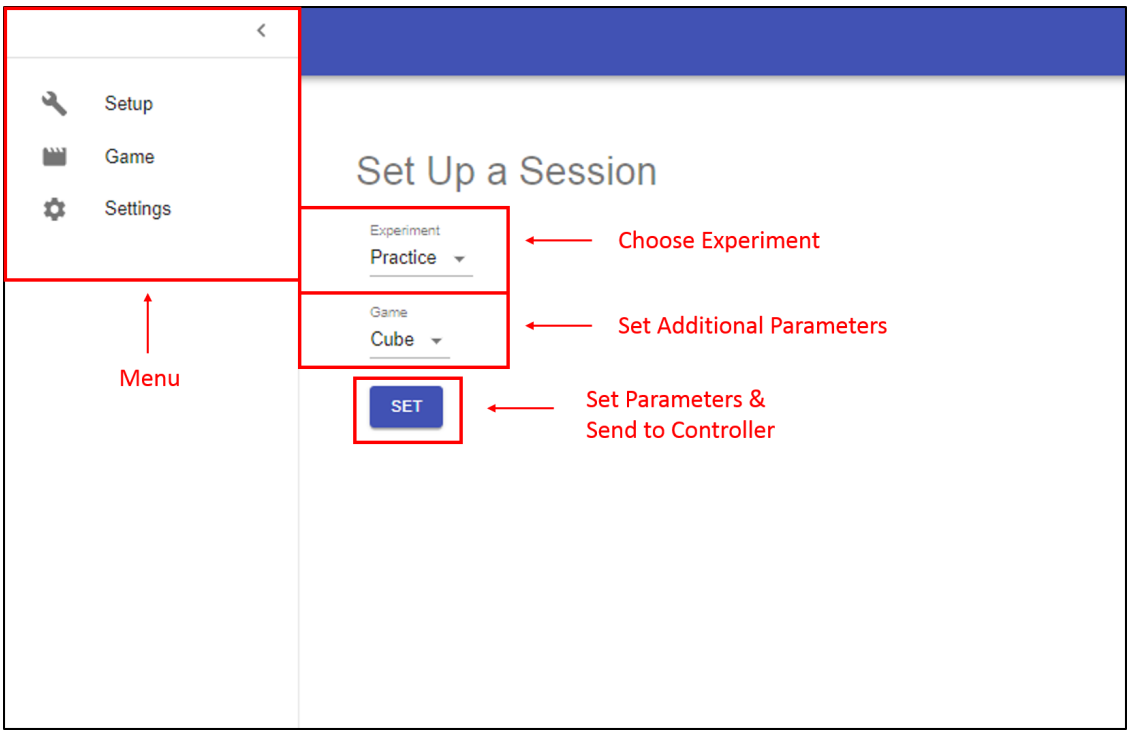

Figure 3.6: The Setup Page

Visuals

The guide is used to display the length of the actuator, the user's position and the desired position (Fig. 3.7). The desired position is shown as a translucent box, which is normally red but changes to green if the user's position is within a certain small margin of error. The User's position is shown as a blue rectangle. In essence, the knee flexion exercise is carried out by the user attempting to follow the desired position as closely as possible.

Three simple games were created, each corresponding to a different type of game which could be expanded upon by future students. The first is a simple cube, displayed with the guide. If the user is within a certain margin of error, the cube spins, otherwise the cube is still. This is meant to demonstrate a virtual environment which responds to the performance of the user, but is not a competitive or challenging game.

The second game is a race between three object: the user and two computer controlled characters (Fig. . Each object moves around a track, with the computer characters moving at

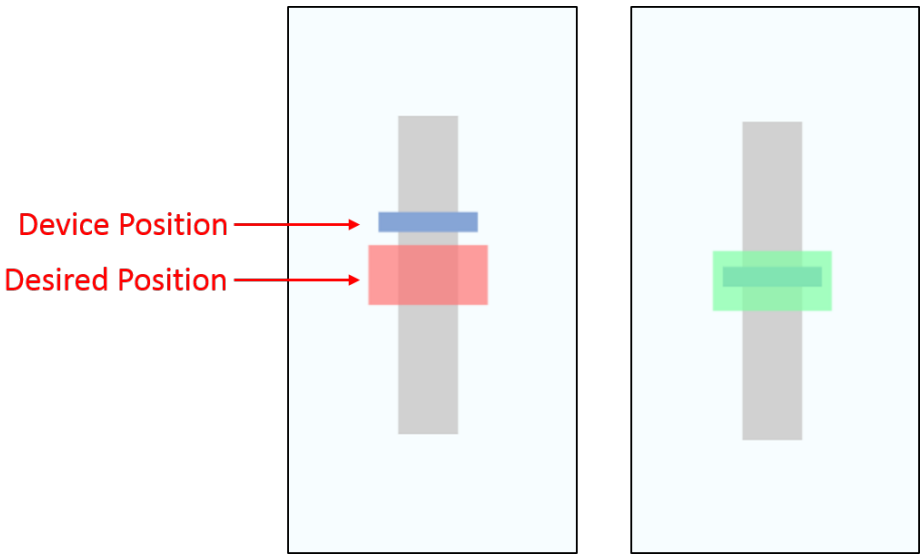

Figure 3.7: The trajectory guide

predefined speeds, and the user's character moving at a speed proportional to the accuracy of their motion (*i.e.* the lower the error, the faster they move). This game is meant to demonstrate a more challenging game which motivates the player to perform the exercise accurately.

The third game is balancing game which is not related to the knee flexion exercise. The user's foot directly controls the position of their character. They are tasked with maintaining a static position on top of a column (Fig REF). Disturbances are launched at the player in the form of spheres, which when in contact created haptic force feedback according to the physics layed out in Sec. REF. The user must push these disturbances away without falling off the column. This game demonstrates both haptic feedback, and a game in which the user's foot directly control a character.

3.6 Software Extension

While the software was customized to this system, it is desirable that it can be re-used in future design changes to the device, such as changing the control algorithm, adding DOF's, or changing the UI. To facilitate the use of the software for future designs, the software

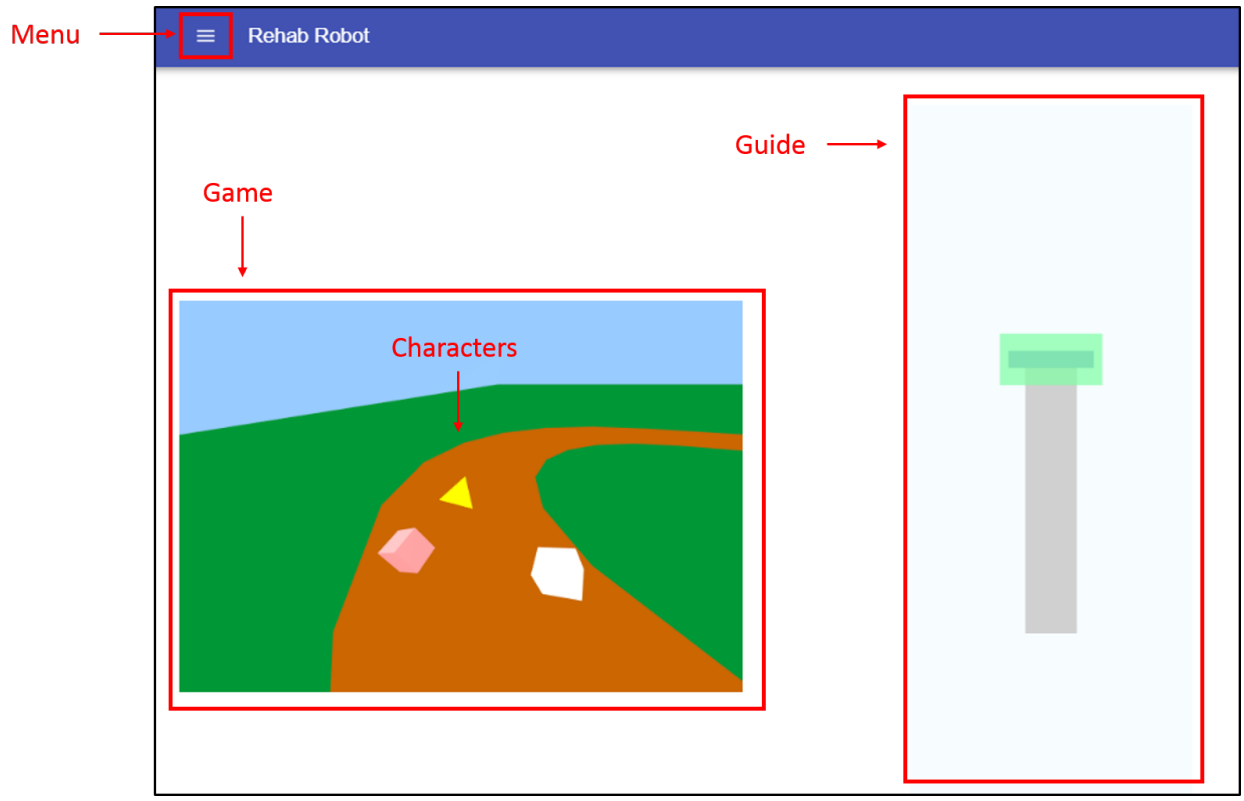

Figure 3.8: The racing game

was re-factored to a more general structure, essentially a software template. The template includes all initialization steps for creating real-time threads, communication of the UI, and communication with the DAQ – however the controller and data structures are empty. Instructions are included on adding ...

Chapter 4

Experiments

4.1 Safety Tests

4.2 Functional Tests

4.2.1 Verifying Admittance Control

The goal of admittance control is to achieve a desired dynamic behaviour, usually in terms of a mass-spring-damper system, given some input force. To confirm that the controller is successfully rendering the desired admittance, the output position of the device was compared to a simulation of the desired admittance. Force was applied to the system, and data regarding the position and the input force was saved. The force data was used as an input to a Simulink simulation of the mass-spring-damper system, and both the robots position and the simulated position were compared. It was expected that the robot position would track the simulation with some minimal delay due to time lag introduced by the finite sampling rate, as well as some possible error due to limitations of the robots PD control.

For the first test no interaction force was applied, only an initial offset of the reference position of the spring (of 0.2m) was used to elicit a response. A comparison of the robot's behaviour and the simulated behaviour of the desired admittance behaviour is given in (). Force was applied to the robot in the second test, in order to verify that the controller

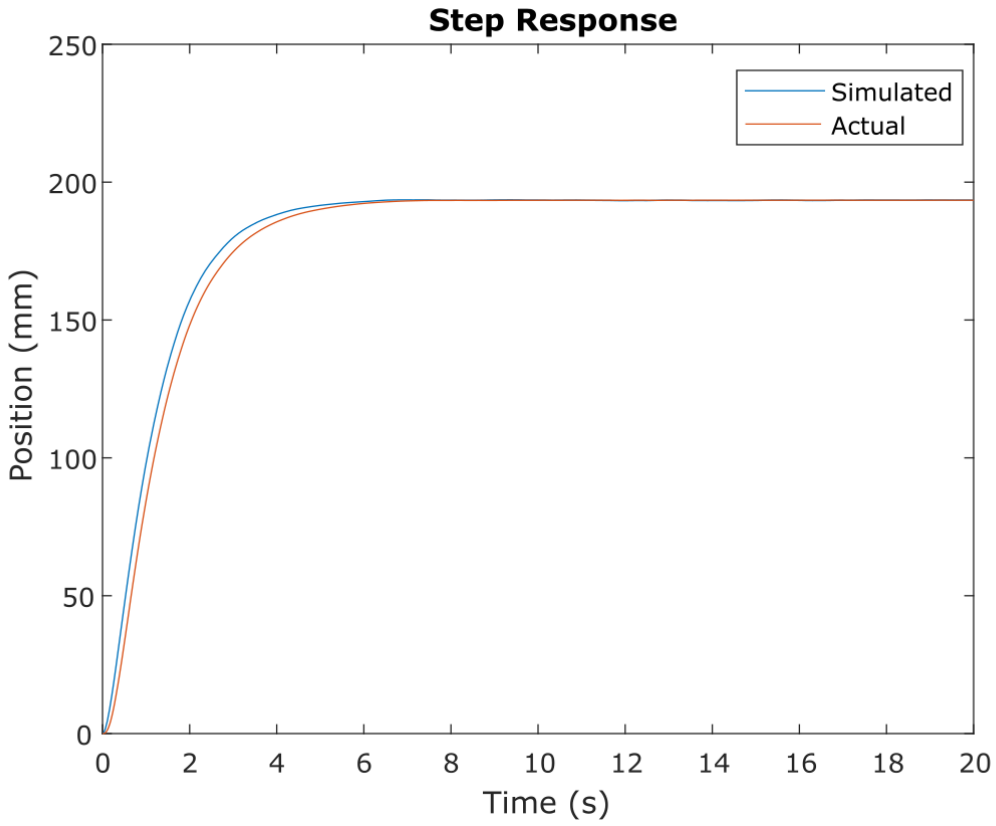

Figure 4.1: Admittance Verification with Interaction

reproduced the desired admittance when interacting with an unknown environment. ...

These tests confirm that the admittance controller adequately recreated the desired dynamics, and that the outer-loop PD controller tracked this behaviour.

4.2.2 Verifying Haptic Control

4.3 Healthy Subject Experiments

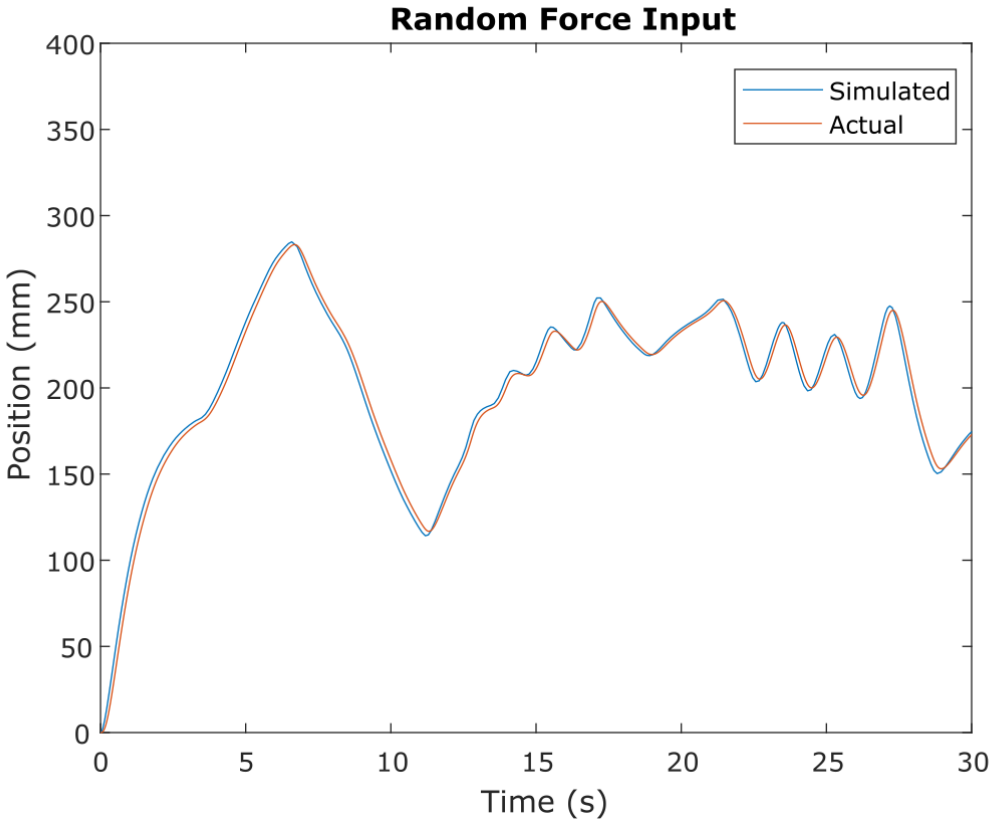

Figure 4.2: Admittance Verification with Interaction

Chapter 5

Conclusion

5.1 Future Work

Appendix I: CAD Drawings

Appendix II: PCB Diagram & BOM

Appendix III: Hardware Datasheets

Appendix IV: Complete Data from Fieldwork at Hospital

Bibliography

[1] E. J. Benjamin, C. S. Salim Virani, C.-V. W. Chair Clifton Callaway, A. M. Chamberlain, M. R. Alexander Chang, M. Susan Cheng, M. E. Stephanie Chiuve, S. Mary Cushman, F. N. Francesca Delling, M. Rajat Deo, M. D. Sarah de Ferranti, F. F. Jane Ferguson, M. Fornage, F. Cathleen Gillespie, M. R. Carmen Isasi, F. C. Monik Jiménez, S. Lori Chaffin Jordan, S. E. Judd, M. Daniel Lackland, F. H. Judith Lichtman, F. Lynda Lisabeth, F. Simin Liu, F. T. Chris Longenecker, P. L. Lutsey, F. S. Jason Mackey, D. B. Matchar, K. Matsushita, F. E. Michael Mussolino, F. Khurram Nasir, F. O. Martin, L. P. Palaniappan, F. Ambarish Pandey, D. K. Pandey, F. J. Mathew Reeves, M. D. Ritchey, M. J. Carlos Rodriguez, F. A. Gregory Roth, F. D. Wayne Rosamond, F. K. Uchechukwu Sampson, F. M. Gary Satou, F. H. Svati Shah, F. L. Nicole Spartano, D. L. Tirschwell, C. W. Tsao, M. H. Jenifer Voeks, J. Z. Willey, M. T. John Wilkins, F. H. Jason Wu, F. M. Heather Alger, S. S. Wong, and F. Paul Muntner, "Heart Disease and Stroke Statistics-2018 Update A Report From the American Heart Association," *Circulation*, vol. 137, pp. 67–492, 2018.

[2] D. Hebert, M. P. Lindsay, A. McIntyre, A. Kirton, P. G. Rumney, S. Bagg, M. Bayley, D. Dowlatshahi, S. Dukelow, M. Garnhum, E. Glasser, M.-L. Halabi, E. Kang, M. MacKay-Lyons, R. Martino, A. Rochette, S. Rowe, N. Salbach, B. Semenko, B. Stack, L. Swinton, V. Weber, M. Mayer, S. Verrilli, G. DeVeber, J. Andersen, K. Barlow, C. Cassidy, M.-E. Dilenge, D. Fehlings, R. Hung, J. Iruthayarajah, L. Lenz, A. Majnemer, J. Purtzki, M. Rafay, L. K. Sonnenberg, A. Townley, S. Janzen, N. Foley, and R. Teasell, "Canadian stroke best practice recommendations: Stroke rehabilitation

practice guidelines, update 2015," *International Journal of Stroke*, vol. 11, pp. 459–484, jun 2016.

[3] S. Ewart, M. Langfeld Layout, R. Sadki, D. Evans, W. Savedoff, A. Singh, B. Stilwell, W. Van Lerberghe, E. Villar Montesinos, P. Banati, M. Beusenberg, S. Colombo, C. Dora, J. Dzenowagis, H. Fogstad, E. Gajraj, G. Galea, C. Garcia Moreno, Y. Hemed, A. Hinman, A. Kalache, R. Kawar, M. Levin, A. Lopez, A. Mechbal, L. Rago, S. Saxena, P. Setel, C. Shahpar, H. Troedsson, A. Yang, D. Bayarsaikha, S. Begg, C. Bernard, D. Chisholm, S. Ebener, E. Gakidou, Y. Guigoz, P. Hernández, M. Hogan, K. Iburg, C. Indikadahena, M. Inoue, K. Lunze, D. Ma Fat, T. Mwase, F. Naville, J.-P. Poullier, C. Rao, D. Rhoades, H. Salehi, J. Salomon, A. Sousa, R. M. Suarez-Berenguela, U. Than Sein, N. Tomijima, N. Van de Maele, S. Volkmuth, and H. Xu, *The World Health Report*. World Health Organization, 2003.

[4] P. Langhorne, F. Coupar, and A. Pollock, "Motor recovery after stroke: a systematic review," *The Lancet Neurology*, vol. 8, no. 8, pp. 741–754, 2009.

[5] G. Rey, A. Donnan, M. Fisher, M. Macleod, and S. M. Davis, "Stroke," tech. rep., 2008.

[6] S. Prabhakaran, I. Ruff, R. A. Bernstein, T. T, L. PJ, and K. M, "Acute Stroke Intervention," *JAMA*, vol. 313, p. 1451, apr 2015.

[7] "Stroke [online]."

[8] B. H. Dobkin, "Strategies for stroke rehabilitation," *The Lancet Neurology*, vol. 3, no. 9, pp. 528–536, 2004.

[9] Z. Warraich and J. A. Kleim, "Neural Plasticity: The Biological Substrate For Neurorehabilitation," *PM&R*, vol. 2, pp. S208–S219, dec 2010.

[10] L. K. Casaubon, J.-M. Boulanger, E. Glasser, D. Blacquiere, S. Boucher, K. Brown, T. Goddard, J. Gordon, M. Horton, J. Lalonde, C. LaRivière, P. Lavoie, P. Leslie, J. McNeill, B. K. Menon, B. Moses, M. Penn, J. Perry, E. Snieder, D. Tymianski,

N. Foley, E. E. Smith, G. Gubitz, M. D. Hill, and P. Lindsay, "Canadian Stroke Best Practice Recommendations : Acute Inpatient Stroke Care Guidelines, Update 2015," *International Journal of Stroke*, vol. 11, pp. 239–252, feb 2016.

[11] AVERT Trial Collaboration group, "Efficacy and safety of very early mobilisation within 24 h of stroke onset (AVERT): a randomised controlled trial.," *Lancet (London, England)*, vol. 386, pp. 46–55, jul 2015.

[12] K. R. Lohse, C. E. Lang, and L. A. Boyd, "Is More Better? Using Metadata to Explore DoseResponse Relationships in Stroke Rehabilitation," *Stroke*, vol. 45, pp. 2053–2058, jul 2014.

[13] C. E. Lang, J. R. MacDonald, D. S. Reisman, L. Boyd, T. Jacobson Kimberley, S. M. Schindler-Ivens, T. G. Hornby, S. A. Ross, and P. L. Scheets, "Observation of Amounts of Movement Practice Provided During Stroke Rehabilitation," *Archives of Physical Medicine and Rehabilitation*, vol. 90, pp. 1692–1698, oct 2009.

[14] J. Bernhardt, H. Dewey, A. Thrift, and G. Donnan, "Inactive and Alone Physical Activity Within the First 14 Days of Acute Stroke Unit Care," *Stroke*, 2004.

[15] A. King, A. McCluskey, and K. Schurr, "The Time Use and Activity Levels of Inpatients in a Co-located Acute and Rehabilitation Stroke Unit: An Observational Study," *Topics in Stroke Rehabilitation*, vol. 18, pp. 654–665, oct 2011.

[16] G. Colombo, M. Wirz, and V. Dietz, "Driven gait orthosis for improvement of locomotor training in paraplegic patients," tech. rep., 2001.

[17] G. Mchugh and I. D. Swain, "A comparison between reported and ideal patient-to-therapist ratios for stroke rehabilitation," *Health*, vol. 5, no. 6A2, pp. 105–112, 2013.

[18] R. S. Calabrò, A. Cacciola, F. Bertè, A. Manuli, A. Leo, A. Bramanti, A. Naro, D. Milardi, and P. Bramanti, "Robotic gait rehabilitation and substitution devices in neurological disorders: where are we now?," *Neurological Sciences*, vol. 37, pp. 503–514, apr 2016.

[19] W. H. Chang and Y.-H. Kim, "Robot-assisted Therapy in Stroke Rehabilitation.," *Journal of stroke*, vol. 15, pp. 174–81, sep 2013.

[20] J. Mehrholz and M. Pohl, "Electromechanical-assisted gait training after stroke: A systematic review comparing end-effector and exoskeleton devices," *Journal of Rehabilitation Medicine*, vol. 44, no. 3, pp. 193–199, 2012.

[21] J. Hidler, D. Nichols, M. Pelliccio, K. Brady, D. D. Campbell, J. H. Kahn, and T. G. Hornby, "Multicenter Randomized Clinical Trial Evaluating the Effectiveness of the Lokomat in Subacute Stroke," *Neurorehabilitation and Neural Repair*, vol. 23, pp. 5–13, sep 2008.

[22] S. K. Banala, S. K. Agrawal, and J. P. Scholz, "Active Leg Exoskeleton (ALEX) for Gait Rehabilitation of Motor-Impaired Patients," in *2007 IEEE 10th International Conference on Rehabilitation Robotics*, pp. 401–407, IEEE, jun 2007.

[23] J. Veneman, R. Kruidhof, E. Hekman, R. Ekkelenkamp, E. Van Asseldonk, and H. van der Kooij, "Design and Evaluation of the LOPES Exoskeleton Robot for Interactive Gait Rehabilitation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 15, pp. 379–386, sep 2007.

[24] K. Y. Nam, H. J. Kim, B. S. Kwon, J.-W. Park, H. J. Lee, and A. Yoo, "Robot-assisted gait training (Lokomat) improves walking function and activity in people with spinal cord injury: a systematic review," *Journal of NeuroEngineering and Rehabilitation*, vol. 14, p. 24, dec 2017.

[25] H. Schmidt, S. Hesse, R. Bernhardt, and J. Krüger, "HapticWalker—a novel haptic foot device," *ACM Transactions on Applied Perception*, vol. 2, pp. 166–180, apr 2005.

[26] S. Hesse, A. Waldner, and C. Tomelleri, "Innovative gait robot for the repetitive practice of floor walking and stair climbing up and down in stroke patients," *Journal of NeuroEngineering and Rehabilitation*, vol. 7, p. 30, dec 2010.

[27] S. Hesse, D. Uhlenbrock, and K. Berlin, "A mechanized gait trainer for restoration of gait," Tech. Rep. 6, 2000.

[28] C. Schmitt, P. Métrailler, A. Al-Khodairy, R. Brodard, J. Fournier, M. Bouri, and R. Clavel, "THE MOTION MAKER : A REHABILITATION SYSTEM COMBINING AN ORTHOSIS WITH CLOSED-LOOP ELECTRICAL MUSCLE STIMULATION," 2004.

[29] M. Bouri, B. Le Gall, and R. Clavel, "A new concept of parallel robot for rehabilitation and fitness: The Lambda," in *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2503–2508, IEEE, dec 2009.

[30] K. J. Chisholm, K. Klumper, A. Mullins, and M. Ahmadi, "A task oriented haptic gait rehabilitation robot," *Mechatronics*, vol. 24, no. 8, pp. 1083–1091, 2014.

[31] K. Chisholm, *Design and Control for a Gait Rehabilitation Robot*, vol. MR71524. 2010.

[32] W. Meng, Q. Liu, Z. Zhou, Q. Ai, B. Sheng, and S. S. Xie, "Recent development of mechanisms and control strategies for robot-assisted lower limb rehabilitation," *Mechatronics*, vol. 31, pp. 132–145, oct 2015.

[33] N. Hogan, "Impedance Control: An Approach to Manipulation," *American Control Conference, 1984 IS - SN - VO -*, no. March, pp. 304–313, 1985.

[34] K. Laver, S. George, S. Thomas, J. E. Deutsch, and M. Crotty, "Virtual reality for stroke rehabilitation: an abridged version of a Cochrane review.," *European journal of physical and rehabilitation medicine*, vol. 51, pp. 497–506, aug 2015.

[35] J. Patel, G. Fluet, A. Merians, Q. Qiu, M. Yarossi, S. Adamovich, E. Tunik, and S. Massood, "Virtual reality-augmented rehabilitation in the acute phase post-stroke for individuals with flaccid upper extremities: A feasibility study," in *2015 International Conference on Virtual Rehabilitation (ICVR)*, pp. 215–223, IEEE, jun 2015.

[36] G. Saposnik, R. Teasell, M. Mamdani, J. Hall, W. McIlroy, D. Cheung, K. E. Thorpe, L. G. Cohen, and M. Bayley, "Effectiveness of Virtual Reality Using Wii Gaming Technology in Stroke Rehabilitation," *Stroke*, vol. 41, no. 7, 2010.

[37] T. T. Jiang, Z. Q. Qian, Y. Lin, Z. M. Bi, Y. F. Liu, and W. J. Zhang, "Analysis of virtual environment haptic robotic systems for a rehabilitation of post-stroke patients," in *2017 IEEE International Conference on Industrial Technology (ICIT)*, pp. 738–742, IEEE, 2017.

[38] E. Vogiatzaki and A. Krukowski, "Serious Games for Stroke Rehabilitation Employing Immersive User Interfaces in 3D Virtual Environment," *Journal of Health Informatics*, vol. 6, nov 2014.

[39] R. Adams and B. Hannaford, "Stable haptic interaction with virtual environments," *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 465–474, jun 1999.

[40] D. A. Lewine, "POSIX Programmer's Guide," tech. rep., 1991.

[41] Sebastian Siewior, "Real-Time Linux Wiki," 2019.

[42] K. M. Obenland, "The Use of POSIX in Real-time Systems, Assessing its Effectiveness and Performance," tech. rep.

[43] The GTK Team, "The GTK Project," 2019.