Title Page

Cover Page

Acknowledgements

Abstract

Table of Contents

Table of Figures

Table of Tables

Table of Equations

1. Thesis definition and scope (6.5 pg)

1.1 Topic Definition

A powered exoskeleton, or exoskeleton, is wearable technology the amplifies and augments the pilot's physicality. Through direct mechanical assistance via actuators, the pilot's effective strength may be increased. By supplementing the strength required to complete a task the energy requirements of the task may be reduced; effectively increasing the pilot's endurance. Possible uses for exoskeletons include: military operations, emergency & rescue, physical/manual labour, and medical applications.

Two major factors impact the viability of exoskeleton technology: power supply, and control. This thesis shall address one facet of the difficulties of exoskeleton control. Current exoskeleton control methods are inadequate due to mechanical constraints and the limitations of the control methods. Imperfections in mechanical design may result in a limited range of movement affecting the suits utility (e.g. A rigid spine in a confined space). Current methods of control use either force-based sensors or preprogramed movements. Finite sets of preprogramed movements are insufficient for dynamic environments (Charara, 2015). Force based methods encounter stability problems and may increase the exertion required to complete a task (Keller, 2016).

This thesis will focus on the development of a novel power exoskeleton control method based on detecting the pilot's position relative to the suit to maintain a constant offset; specifically focusing on the development of the controls and perception systems required to direct an exoskeleton.

By maintaining a constant offset from the pilot, the exoskeleton exists as a concentric outline (or *bubble*) of the pilot, mirroring their actions. To control the system the pilot simply needs to assume the desired position of the suit. By mimicking the pilot's actions, the suit is more intuitive that force based and preprogramed methods. With no physical contact required to operate the system, the energy required from a pilot to complete a task with a load is effectively the same as completing the task with no load. Therefore, with any arbitrary load the pilot has the endurance to perform the task as if there no load at all.

For discussing the structure of the exoskeleton, the term link shall be used.

> ***Link**: Actuated limb segments of the exoskeleton correlating with the thigh (femur), shin (tibia), and foot (metatarsals).*

1.2 Hypothesis

For controlling the suit, it may be assumed that the pilot is inside the suit during operation and the pilots desired position for the suit may be treated as their position. Under these conditions the error between the desired configuration and the actual configuration of the exoskeleton is the difference between the configuration of the pilot and the configuration of the exoskeleton

Therefore, the suit can be controlled accurately (that is to say, error can be known at any time) by observing the position of the pilot relative to the suit. **It is proposed to develop a proof of concept for a positional exoskeleton control system based on measurement of the pilot's position/proximity the suit**. By maintaining a constant offset from the pilot, the exoskeleton may exist as a concentric outline (or *bubble*) of the pilot, mirroring their actions.

In a circumstance where the exoskeleton encounters an obstacle it is desirable to regulate and control the force output of the system. It is desirable to decouple the control of force output and speed (a noted flaw with force-based control methods). To ensure safe movement that does not apply undue force to the environment, the force output of the exoskeleton should be measured and regulated at external contact points.

If the system applies force up to a safe maximum, then once that maximum is met then the exoskeleton will stop matching pace with the pilot's movement and the constant offset between the pilot and the system will not be maintained. The pilot then may contact the internals of the exoskeleton. It is possible to use the pilot's continuing attempts to towards the obstacle as intent to increase force output of the suit.

By measuring the force applied to external and internal contact points by the exoskeleton and the pilot respectively it is possible for the exoskeleton to operate with safe low force outputs which a pilot may override when increased force output is desired.

For the proof of concept, the difference in force applied to internal and external contact points shall be used to control the force output of the exoskeleton.

The subsequent system in summary:

- Uses position sensors to determine the desired configuration of the exoskeleton from the bodily configuration of the pilot;
- Uses external sensors to regulate the force output of the system, maintaining a safe maximum; and,
- Measures force applied internally to determine the force output of the system.

The potential benefits of such a system are dynamic and intuitive controls with effortless operation.

1.2.1.1 Dynamic control

By mirror the movements of the pilot, with a sufficient mechanical design, the system is only limited by the capabilities of the pilot. Therefore, in any system which a human could navigate the exoskeleton should be able to operate. Compared to preprogramed systems, it will be possible to navigate uneven terrain, switch contexts, and perform in unpredictable environments.

1.2.1.2 Intuitive control

The system described shall provide more intuitive control relative to other solutions. If the pilot seeks to move the left leg of the system, they must simply move their left leg. If the suit contacts an object the exoskeleton will cease movement. If the pilot wishes to push the object, they simply need to push the object through the suit. The pilot may control the exoskeleton as they would their own body.

1.2.1.3 Effortless operation

The system significantly increases the effective endurance of the pilot while requiring no exertion to use. With a sufficiently strong exoskeleton, the system may be loaded with any arbitrary weight, but the effort required by the pilot to walk will remain unchanged. The exoskeleton effectively gives the operator carrying a load the endurance of an operator with no load. Note, the magnitude of this benefit increases as the load increases.

1.2.1.4 Functional Requirements

To determine the viability of the exoskeleton and develop a proof of concept it is essential to define the required capabilities of such a system. The following outlines the requirements for a functional exoskeleton system:

1. Steady-state/static operation;
2. Dynamic and actuated operation with non-regulated/imprecise force application;
3. Dynamic and actuated operation with regulated force application; and,
4. Dynamic and actuated operation with regulated force application under real-time conditions.

Should the system be capable of achieving level 4 operation it can be said to be fully functional. To assess the system’s level of functionality specific test case are required which may be considered representative movements of the requirements of each level of functionality. These are outlined as follows in Table 1 and 11 - Appendix A: Representative Movements[1].

Table 1: Functional levels and associated movements

Functionality Level	Representative Movement	Position of Pilot	Position of Exoskeleton	Force Applied by Pilot	Force Applied by Exoskeleton
L1	Standing	✓	✓		
L2	Squatting	✓	✓		
L4	Sitting	✓	✓	✓	✓
L5	Sprinting	✓	✓	✓	✓

As seen in Table 1, there are four main pieces of information required to control the exoskeleton at all levels:

- Position of pilot (relative to exoskeleton);
- Position of exoskeleton (relative to world);
- Force internally applied by pilot; and,
- Force externally applied by exoskeleton/

Note that every representative movement can be completed using only the lower extremities. This implies that to develop a proof of concept for exoskeleton only a lower extremity exoskeleton is required.

[1] Climbing stairs was originally include in the representative movement. It is still included in the appendix but is not *strictly* required to demonstrate functionality.

1.3 Scope

1.3.1 Proof of Concept

The purpose of this thesis is to develop some of the major subsystems for a proof concept for a position-based exoskeleton control system. As noted above in 1.2.1.4 - Functional Requirements, to create a proof of concept for the system only a lower extremity exoskeleton is required.

1.3.1.1 Task Division

Creating said proof of concept was beyond the scale and scope of a single undergraduate thesis. Instead, the task was to be divided amongst two students, who would complete subsystems independently before integrating their work. It was determined that the most functional demarcation of tasks would be to divide the system according to "determining the required actions" and "performing the required actions". Broadly speaking, one student would design and create the sensing/perceiving and control systems for the proof of concept, and the other would create the structural and actuation systems of the proof of concept[2]. The point of integration between the two systems would be a communication system capable of transmitting the desired action from one side to the other.

This student was assigned the perception and control systems[3].

1.3.1.2 Inclusions (In Scope)

Based on the specific division of tasks and the demarcation devised the following major functional requirements were identified as within scope:

1. **Detection of Pilots position relative to the exoskeleton (detection of the suit's absolute position would be the responsibility of the actuation system).**
   1.1. This includes the hardware, firmware, software, and mechanical structure required.
   1.2. This is limited to detection of the position of the femur, tibia, and foot (treated as a singular entity). This does not include the detection of the position of individual toes or the internal actuation of the foot.
2. **Force application of the exoskeleton to the environment and the pilot to the exoskeleton**

[2] As such one student would be responsible for determining the required action from the exoskeleton systems to perform as desired, and one student would create a system that was capable of performing said actions

[3] Actuation system, control system, perception system, and structural system are descriptive terms for the approximate scope and manner of certain groups of subsystems. They are not prescriptive and should not be treated as such, e.g. the mechanical structure required to hold the force sensors in place is structural but is within the scope of the perception systems not the structural system.

2.1. This includes the hardware, firmware, software, and mechanical structure required.

2.2. This is limited to the detection of force application at the soles of the feet and the rear of the pilot, zones required for the representative movements.

2.3. This is limited to a rigid sole without actuation, i.e. the foot may move and bend at the ankle but shall not be treated as flexing at the ball of the foot.

3. **Control system for determining required action (torque) from actuation system for correct operation**

   3.1. This is limited to determining the desired torque and angle of the actuation systems.

   3.2. This does not include determining power, voltage, or current requirements for actuators.

   3.3. This does not include determining control inputs (e.g. pulse width modulation duty cycles) for the actuation systems.

4. **Communication from control & perception software to actuation system**

   4.1. This is limited to creating an input and output connection for interfacing with the actuation & structural system via a common protocol.

   4.2. This does not include the implementation of a communication protocol for the student responsible for the actuation & structural systems[4].

1.3.1.2.1 Variations

The original scope of the project was extended to include the design and create of a simplified actuation system capable of refining, testing, tuning, and demonstrating the controls and actuation systems[5]. Consequentially, a fifth major function requirement was added to the system:

5. **Development of actuation system sufficient to demonstrate attainment of other major function requirements.**

   5.1. Commissioning of actuators and mechanical structure required to demonstrate functionality of position detection systems;

   5.2. Commissioning of actuators and mechanical structure required to demonstrate functionality of force detection systems;

[4] Each student was responsible for their own firmware and communications implementation.

[5] The original proposed scope did not include the creation or design of any actuations, or the interfacing between the control and perceptions systems and the systems actuators. As the project progressed it became apparent that the mechanical/actuation section of the project would not be completed in time for operation and that to develop and demonstrate the functionality of the controls and perception system a testing rig would be required.

5.3. Development of motor interface and power systems required to control actuators in the desired fashion.

1.3.1.3 Exclusions (Out of Scope)

The following tasks were considered out of scope and where excluded from the project:

- Commissioning of the torso, head, or upper extremities of an exoskeleton;
- Measurement of actuator positions or absolute exoskeleton position;
- Measurement of velocity, acceleration, or torque of any section of the exoskeleton;
- The development of an exoskeleton capable of supporting additional loads, i.e. carrying weights beyond those required for demonstration of proof of concept;
- There was no compensation for the flexion and distortion of body parts, e.g feet;
- Addressing power consumption problems, power-to-weight ratio problems, or price problems associated with exoskeletons;
- Actuation points (hip, knee, ankle) where constrained to 1 degree of freedom (DOF); and,
- Anything not in scope.

2. Background & Problem Breakdown (13.5 pg)

2.1 Background & Prior Art

Exoskeleton technology began in 1890 (United States of America Patent No. 440684, 1890), with Nicholas Yagin's development of a passive device that used compressed gas to assist in human movement.

However, it was not until the 1960s that the first prototypical powered exoskeleton was developed. The Hardiman (Keller, 2016) shown in Figure 1,created by General Electric, was non-viable due to its extreme weight[6] and control problems. The suit, when used as a complete system instead of in parts, was subject to dangerous violent uncontrolled movements and the master-slave control system suffered debilitating lag.

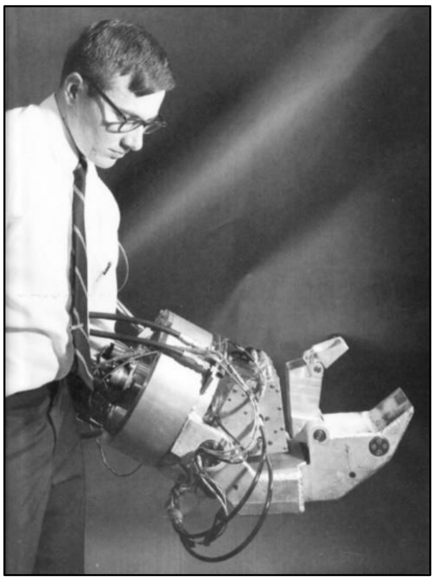


Figure 1: G.E. Hardiman I Exoskeleton (Cybernetic Zoo, 2010)

Prospective uses for exoskeletons usually involve a scenario where a human pilot may require the strength and endurance of a machine, but wheeled vehicles are undesirable. Examples of possible applications include[7]:

- Military Operations: operators are required to carry heavy loads over longs distances and operate in dynamic and unruly conditions. Difficult terrain, heterodox environments, and general disarray result in wheel machinery often being unsuitable.

[6] Double its maximum load.

[7] These applications represent some of the broader uses for exoskeletons, neglecting the role of specifically designed exoskeletons for niche tasks: shock absorbing legs for parachutes/paratroopers, self-propelled underwater diving suits, etc.

- Rescue and evacuation missions: Similar to military operation with the additional concern of environmental hazards and structural collapse. In the event of a fire or chemical incident, the safety equipment and tools required can be cumbersome; exoskeletons can alleviate some of this burden. Where structures are damaged or collapsed an exoskeleton can provide the extra strength required to save a life,
- Medical Systems: When amputation, age, or illness results in an individual suffering from reduced mobility and strength exoskeletons present exciting opportunities to compensate for their pilot's impediments[8].
- Construction & Physical Labour: Similar to rescue and evacuation missions; equipment and tools can be large, heavy, and cumbersome, and construction sites may have insufficient access for wheel vehicles. Occupations where brute human strength is employed to lift, move, or carry objects can be aided immensely by the application of exoskeletons.

Additionally, certain equipment is bounded by the size of human operators (e.g. firehose, jack hammer, etc…) may be increased, increasing the scale and speed of tasks.

Since the Hardiman, exoskeletons have been plagued by the same two major problems that have prevented their use in real world applications: power to weight ratio/power supply[9] and control.

2.1.1 Preprogramed Control

Preprogramed control methods consist of a set of specific movements that are triggered by the pilot. These systems[10] are inherently limited in their utility. By having a finite or procedurally generated set of movements there will always be scenarios or circumstances where the set of movements is not applicable. In real environments (e.g. military, rescue & evacuation, and physical labour) dynamic controls are required.

As noted by Dunietz when using an exoskeleton with preprogramed controls, the "human does try to join in the motion, the two get in each other's way, cancelling out the gains for all but the most extreme disabilities." (Dunietz, 2017). While preprogramed movements have limited

[8] Medical exoskeletons often compensate for a patient to move without assistance (e.g. paraplegic or stroke survivor). Their goal is often to move *instead of the pilot, not with the pilot* and as such are not the likely beneficiaries of dynamically controlled exoskeletons.
[9] Self-supporting exoskeletons capable of useful tasks require impractical quantities of power. Suits are either feeble, tethered, or have a short battery life
[10] HAL measures contractions in the arms of patients to trigger left-foot right-foot walking motions. Warrior Web applies torque to the ankle of the pilot (assisting them walk) when movement is detected

applicability[11]; For an able-bodied pilot, preprogramed movements are inadequate and “a bit like being a marionette with four wires controlling my legs” (Cornwall, 2015).

2.1.2 Force Based Control

Force based control systems use force applied to the internals of a suit to determine the pilots desired position. The force applied indicates the direction and magnitude of movement. Force based systems are often inadequate for practical applications due to the sensitivity of force input. Systems which are too sensitive may develop jitter, and delays between sensing and movement combines with physical inertia may result in the system applying force to the pilot, creating an unstable feedback loop. Systems with are insensitive are slugging and require the pilot to push and move against the suit. Using these systems can be cumbersome and exhausting to use.

As the only mechanism for detecting position for a force-based system is the pilot making contact with the suit, misalignments in sizing can result physical dead bands when pilots are unable to touch the suit and the control system is effectively blind. Additionally, suits which maintain constant contact with asymmetrical body parts may interpret asymmetry as force input and therefore require constant active resistance from the pilot to control.

Finally, force-based systems do not distinguish between the force output of the system and the speed desired. If a pilot wishes to move quickly they must apply a large amount force to the system, if the suit encounters an obstacle this movement is then interpreted as a large amount of force applied to the object. There is no mechanism for quick safe movements.

For exoskeletons in dynamic real-world environments to be viable improvements on the existing force-based sensing methods are required.

[11] In circumstances where the movement of the pilot is so limited and restricted (e.g. via disability) that any mobility is an improvement.

2.2 Controls

"A control system is an interconnection of components forming a system configuration that will provide a desired system response."
(Ogata, 2010).

A closed loop control system uses feedback and measurements to compare the system with the desired output. By modelling a system and developing the appropriate controls system we may generate the desired behaviour from the system.

A closed loop control system can be summarised as the interaction of (Ogata, 2010):

- Controllers: which are informed of the desired behaviour while receiving feedback from sensors on the actual behaviour of the system;
- Actuators: which modulate their behaviour based on instructions from the controllers;
- A process to be controlled; and,
- Sensors: which measure the behaviour of the system (the actual output of the process) and inform the controllers.

One method of conceptualizing the behaviour of a system is to create block diagrams of the system (Golnaraghi & Kuo, 2010). Block diagrams may be used to communicate comparators, transfer function components, input and output signal, feedback loops, etc. We may describe the state of a closed loop system via block diagrams, see Figure 2.

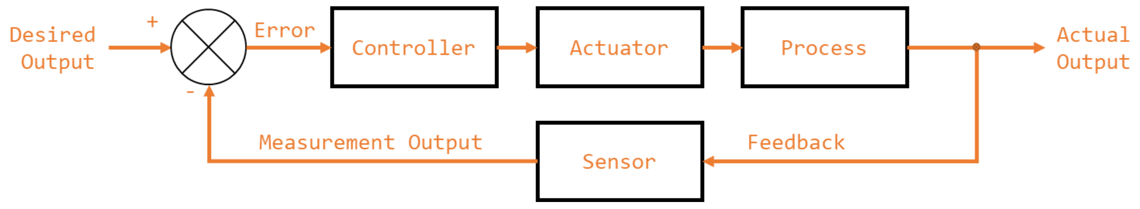


Figure 2: Closed Loop Control System Block Diagram

Here the error of the system is the difference between the desired state and the measured state. The (usual) goal of controls is to reduce the error to zero. This may be accomplished by numerous methods, which include the transfer function method[12].

[12] The root locus method could be used to develop the controls for the system. However, that would just be a recount of Ogata and as such we refer to Modern Control Engineering (Ogata, 2010), rather than repeating it. Chapter 6 addresses design by the Root-Locus Method and Chapter 10 speaks directly regarding servo system design (a perhaps more apt framing of the 'actuators side' of the project).

2.2.1 Transfer Function Approach to Modelling and Control

"The transfer function of a linear, time-invariant differential-equation system is defined as the ratio of the Laplace transform of the output (response function) to the Laplace transform of the input (driving function) under the assumption that all initial conditions are zero." (Otaga, 2004)

The transfer function, $G(s)$ for a system as given by Equation 1; where $X(s)$ is the Laplace transform of the input and $Y(s)$ is the laplace transform of the output. This may be seen diagrammatically in Figure 3. The transfer function allows the mapping from a set of inputs to the outputs of system.

$$G(s) = \frac{X(s)}{Y(s)}$$

Equation 1: Transfer Function

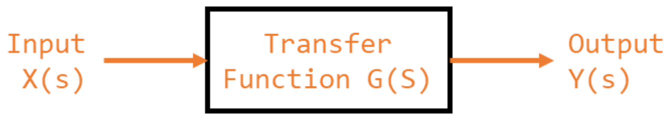


Figure 3: Transfer Function

For a closed loop system, we may say that the behaviour of the system, as seen in Figure 4, may be described by the relationship of the control system and the system process with respect to the error (Golnaraghi & Kuo, 2010).

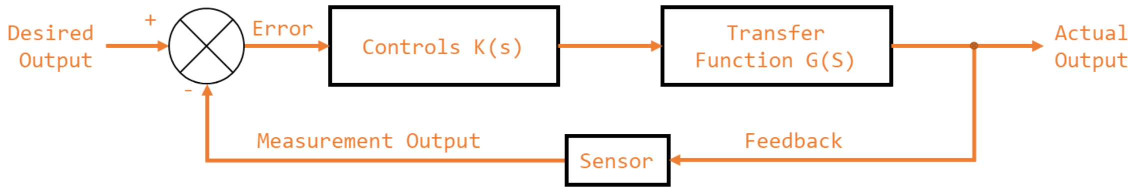


Figure 4: Controlled Closed Loop System

We may minimise the error of the system by correctly curating the control system (K). Given a system described by G(s), by controlling K(s) (the controllers and actuators) we can ensure that the actual output of the system is the desired output of the system (Otaga, 2004). The transfer function for this system, based on the block diagram in Figure 5, is given by Equation 2.

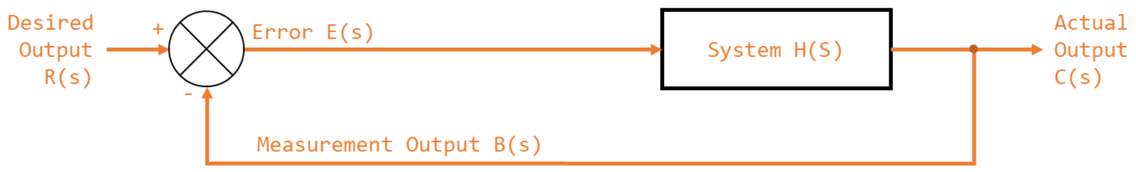


Figure 5: Controlled Closed Loop System (Parameterised)

$$C(s) = E(s)H(s) = E(s)\big(K(s)G(s)\big) = \big(R(s) - C(s)\big)\big(K(s)G(s)\big)$$

$$TF = \frac{C(s)}{R(s)} = \frac{H(s)}{1 + H(s)} = \frac{K(s)G(s)}{1 + K(s)G(s)}$$

Equation 2: Controlled Closed Loop Control System Transfer Function

To find the transfer function for the system it must first be modelled, this is accomplished by finding the equations of motion for the system. Appendix E: Equations of Motion details the process of determining the explicit form of the equations of motion.

2.2.2 Proportional–Integral–Derivative controller

One approach to controls is to amplify the actuators response proportionally to the error. This is known as proportional control and is shown in Figure 6.

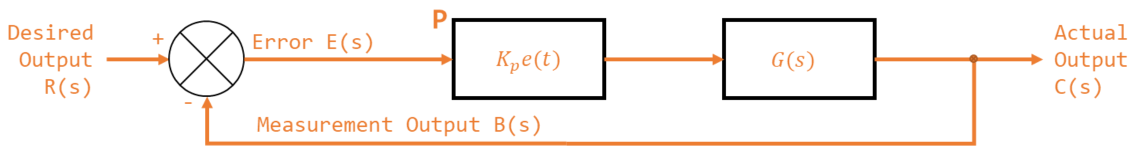


Figure 6: P Control

If the proportional gain of a system is too small the response time of the system may be too slow. If the value is to high the system may overcompensate, resulting in oscillations and overshooting of the target values. If the proportional gain is sufficiently high the system may oscillate so wildly that it becomes unstable (Otaga, 2004). Unstable meaning the system never reaches the desired output and the error of the system increases over time. (National Instruments, 2018).

Proportional control is effective at addressing the gross present error in the system. To create a fast response time for a system, without increasing the proportional gain to unstable values often a derivative term is employed.

Figure 7: PD Control depicts a system with a proportional (K_p) and a derivative gain (K_d). K_d changes in response to the rate at which error is changing. By incorporating a proper K_d value

the system can effective pre-empt change in error, leading to a faster response time (Ogata, 2010).

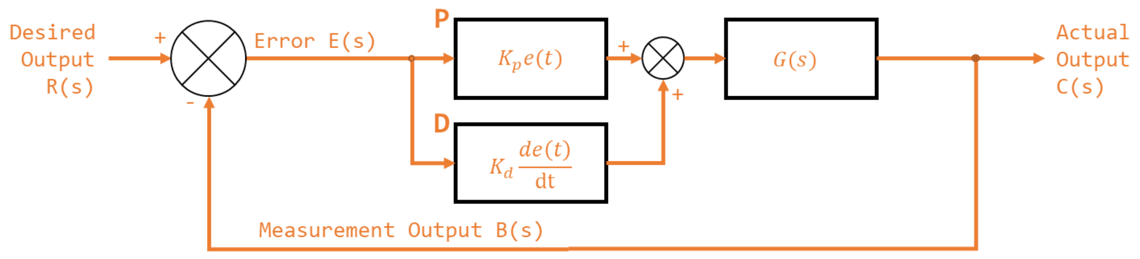


Figure 7: PD Control

A K_d value that is too low will result in a system that doesn't react strongly to changes in error. A K_d value that is correctly calibrated will result in a system that reacts strongly to sudden changes in the system. A K_d value that is too high will reacts too strongly to changes in error and will become highly sensitive to signal noise. Small changes in absolute error which would be ignored by the proportional term may illicit an unwarranted response from an overly sensitive K_d.

For small errors the proportional gain may be insufficient for correction and increasing the proportional gain may result in instability. For constant small errors the change in error may be too small for the derivative gain to correct and increasing the derivative gain may result in instability. Small persistent error in the system, otherwise known as steady state error, may exist in a system with pure PD control.

To compensate and correct the steady state error in a system an integral gain term may be introduced. A system with an integral gain may be seen in Figure 8. The integral gain in practical terms accumulates historical error in the system. This way, even small errors can slowly increase the integral term to an actionable magnitude.

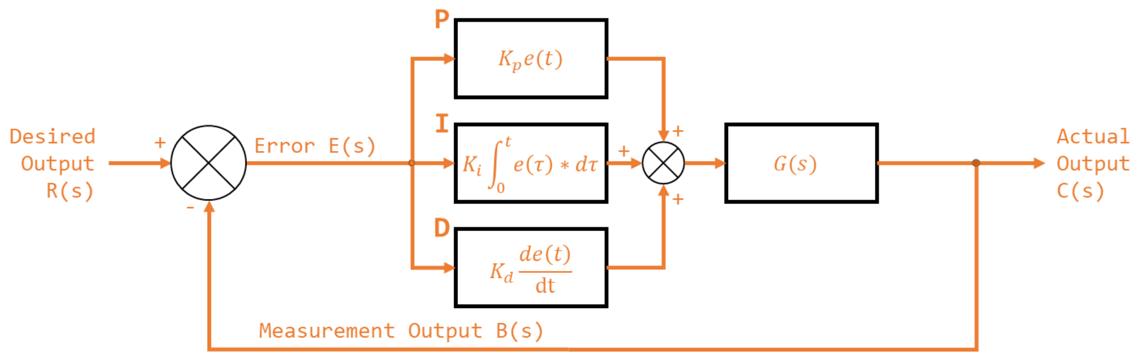


Figure 8: PID Control

An integral gain that is too small will take a very long to correct steady state error within the system. An integral gain that is too large will suffer from integral windup, specifically excessive overshooting the target value. Integral windup is phenomena where a sudden and large change in error cause the integral term to accumulate a massive error that saturates the entire PID response leading to overcompensations. An integral gain that is too large may result in integral wind up of sufficient magnitude that the system becomes unstable.

The transfer function for a PID controller is given by Equation 3.

$$K(s) = K_p + K_d s + \frac{K_i}{s}$$

Equation 3: PID Transfer Function

2.2.3 PID Tuning

There is a plethora of methods for finding the correct range of values for a systems PID to achieve the desired controls[13]. Two were used in this thesis kt (SECOND ORDER):

- Software tools, specifically MATLAB’s PID tuner pidTuner; and,
- The second Ziegler-Nichols method for empirical PID Tuning.

2.2.3.1 Ziegler-Nichols PID Tuning

The second Ziegler-Nichols (ZN) method entails (Ogata, 2010):

1. Initialise the K_d and K_i terms to 0 (Equation 4). Note, that in the literature (Ogata, 2010), K_d and K_i are expressed as the T_d and T_i in terms of the K_p (Equation 5) to generate a control transfer function of Equation 6.

$$K_d = 0 \,\&\, K_i = 0$$

Equation 4: ZN Initial Values

$$T_d = \frac{K_d}{K_p} \,\&\, T_i = K_i K_p$$

Equation 5: ZN Convention

$$K_{\text{Ziegler-Nichols}}(s) = K_p \left(1 + T_d s + \frac{1}{T_i s}\right)$$

[13] Excellent sources include: Automatic Control Systems (Golnaraghi & Kuo, 2010), Modern Control Engineering (Ogata, 2010), Springer Handbook of Robotics (Siciliano & Khatib, 2016), and Modern Control Systems (Dorf & Bishop, 2011).

Equation 6: ZN TF

2. Beginning at 0, increase K_p slowly to the *critical value*, K_{cr}. Where K_{cr} is the lowest K_p “at which the output of the system exhibits sustained oscillation” (Ogata, 2010);
3. Determine the corresponding period (P_{cr}) of K_{cr}; and,
4. Using P_{cr}, K_{cr}, and the values given in Table 2 find the approximate PID values for the system.

Table 2: Ziegler-Nichols Tuning Rules

Controller Type	K_p	T_i	T_d
P	$0.5K_{cr}$	∞	0
PI	$0.45K_{cr}$	$0.833^{\cdot}P_{cr}$	0
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

Note that the Ziegler-Nichols parameters attained are not be taken as absolutes. They are systematically determined estimates of the optimal values. Once found empirical testing should be conducted to refine the values and ensure stability.

2.3 Miscellaneous

2.3.1 Load Cells

Transducers are “elements that convert from one form of energy to another for example, sound to electricity” (Agarwal, 2005). A load cell is a type of transducer that converts the application of force or pressure into voltage or a change in imprudence. A load cell may be effectively used to measure the force applied to a surface, and the fundamental technology behind the common bathroom scale.

2.3.2 Communication

Parallel communication is the method of communicating a multibit message over multiple channels. With enough channels, parallel communication may be used to communicate a message of *any* size in a single clock cycle. However, communication channels can be resource intensive (i.e. using lots of cables and pins, the physical size of the interconnect).

Instead serial communication involves sending data over a single channel sequentially. While often slower than parallel communication, serial communication is often preferable due to the scarcity of I/O lines on microcontrollers. A protocol of encoding messages is used to transmit messages in a consistent intelligible manner.

Serial communication may be synchronous or asynchronous. Synchronous uses a clock signal to synchronise communication. This can result in faster more reliable communication but depends on a centralised clock signal; to maximise demarcation between the project sections, synchronous communication shall not be used.

Asynchronous serial communication may be accomplished by the use of a pair of wires (one for transmission, one for receiving) and by transmitting messages in binary. Messages which can be encoded in binary (e.g. ASCII) may be transmitted in this manner.

A universal asynchronous receiver-transmitter (UART) is a hardware device for asynchronous serial communication that may be integrated into a microcontroller. UART may be used to implement asynchronous serial communication on a microcontroller via I/O lines.

Direct memory access (DMA) is a mechanism by which hardware systems (like UART) are able, independent of the central processing unit (CPU), to interface directly with the main system memory. By bypassing the CPU read and write operations can be completed faster, while the CPU is dedicated to other tasks. For example, messages received via UART may be directly stored in memory, and messages to be sent via UART may be loaded into buffers directly.

2.3.3 Servomotor

A servomotor is a linear or rotary actuator with a closed-loop control system used to manage its behaviour. Usually a motor will be paired with an encoder to provide feedback to the system. The specific of the implementation of servomotors vary, as does their ability to control position, velocity, and acceleration.

2.3.4 Pulse Width Modulation (PWM)

Pulse-width modulation (PWM) is a modulation technique most commonly used to encode and control the power output of motors. PWM works by toggling the output signal to a load sufficiently quickly to approximate analogue behaviour. I.e. in a system where power cannot directly be controlled, PWM can be used to ensure that the output signal is high 50% of the time, effectively demanding 50% power to be supplied. As long as the switching frequency is so high as for the resultant waveform to be perceived as continuous the load will behave as if driven by a continuous supply.

For the control of servomotors PWM is often used to indicate the desired behaviour (indicating position, speed, etc…). The signal delivered to the servomotor will be held low and then toggled high for a duration. The duration of the high period, or pulse, in relation to a full cycle period is the duty cycle. The duty cycle indicates to the servo the desired behaviour[14].

[14] See 8.2.4 - PWM Control

3. Approach and execution (30 pg)

3.1 Definition and Requirements

The system defined by the scope[15] was decomposed into its major function requirements. Functional decompositions were then completed for the major subsystems in their respective sections. This section outlines the major function requirements of the supersystem.

The integrated system for the lower extremity exoskeleton was at all times to maintain a constant offset from the pilot while maintaining proper force application. This was to be accomplished, as seen in Figure 9, by measuring the pilot's position relative to the suit (SS1), measuring force application at the contact points (SS2), deciding what action to perform (SS3), and communicating (SS4) that to the actuation system (SS5). Individual subsystems however, do not constitute a solution; the super system must also be considered.

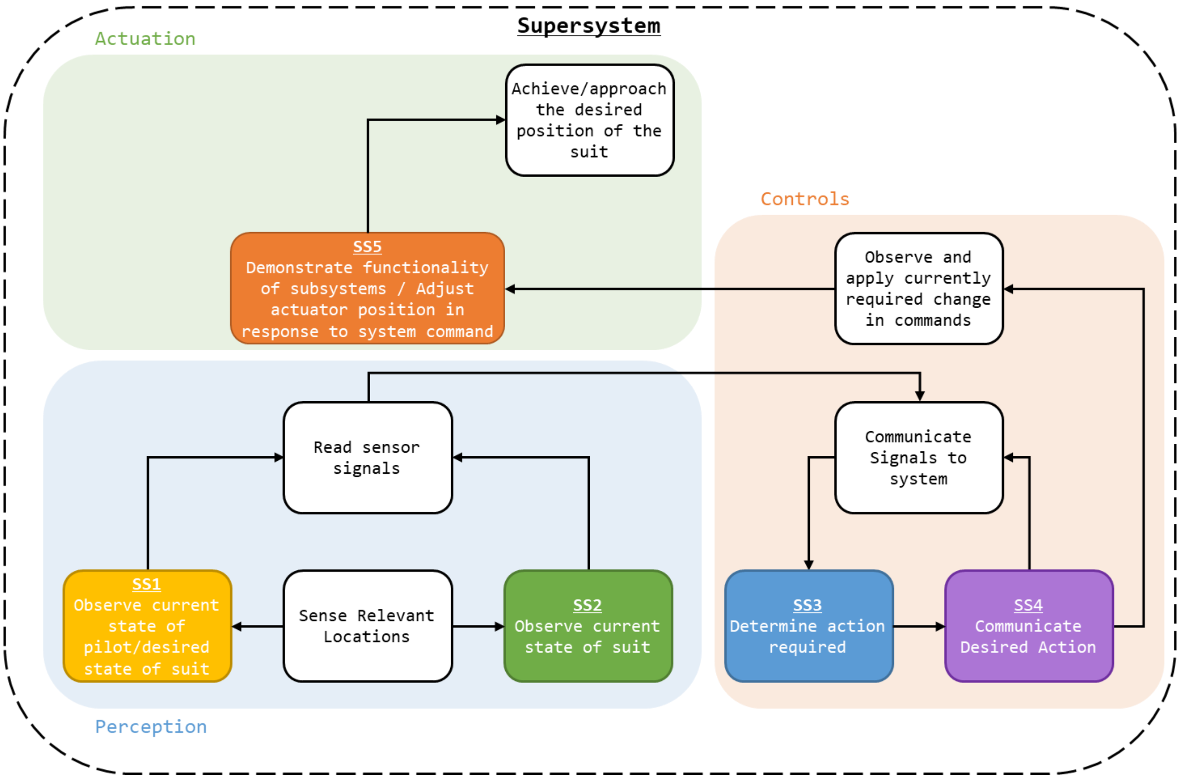

Supersystem
Actuation
Achieve/approach the desired position of the suit
SS5 Demonstrate functionality of subsystems / Adjust actuator position in response to system command
Controls
Observe and apply currently required change in commands
Communicate Signals to system
SS3 Determine action required
SS4 Communicate Desired Action
Read sensor signals
SS1 Observe current state of pilot/desired state of suit
Sense Relevant Locations
SS2 Observe current state of suit
Perception

Figure 9: Supersystem Breakdown

Note that the supersystem takes the form of a closed loop control system with a perception system informing controllers which instruct the actuation system, as shown in Figure 4.

[15] See 1.3 - Scope

The requirements and functions specific to the supersystem[16] are:

- To read the sensor signal values attained by SS1 and SS2;
- To communicate the state of the system (these signal readings) to SS3;
- Send through the commands, via SS4, required of the actuators; and,
- Achieve the desired position of the suit, as actuated by SS5.

3.1.1 Additional Consideration

For the entire design the following additional considerations were evaluated:

- Mass should be minimised where practical[17];
- M3 was standardised as the sizing for screw, nuts, bolts, fittings, etc; and,
- Signal-Vcc-Ground was standardised as cable/header ordering.

3.2 Approach and Execution

3.2.1 Sensor Placement

As noted in 4.2.1 - Perceiving Distance, it is possible to determine the position of the pilot by measuring the relation of a point to a fixed origin. However, proper placement of the sensors is essential. As seen in Figure 10, the further away the sensor array is placed from the rotational axis, the greater the observed changed in distance for a given angle. As such sensors should be placed as far away from their rotational origin as possible.; however, if the hips are to be considered a fixed point[18] then deviation in the position of the pilot and the exoskeleton propagates from the hips. As such measured error will maximised for each link at the end of the link[19] furthest from the pilot.

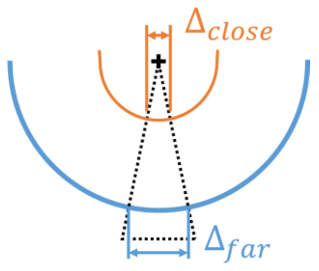


Figure 10: Delta Position vs Distance

[16] Not assigned to a subsystem.
[17] Increasing the mass of any component that is supported by the exoskeleton increases the mass (and inertial term) of the leg. This in turn increase the torque requirements of the actuators. As the system is not to be load bearing, mass should be minimised.
[18] See **Error! Reference source not found. - Error! Reference source not found.**
[19] Links refers to the actuated limb segments of the exoskeleton correlating with the thigh, shin, and foot.

3.2.2 System Configuration

By placing the position-based sensors at the most practical furthest position from pilot, the configuration shown in Figure 11 was created. Force sensors where to be placed at the contact points[20] as depicted.

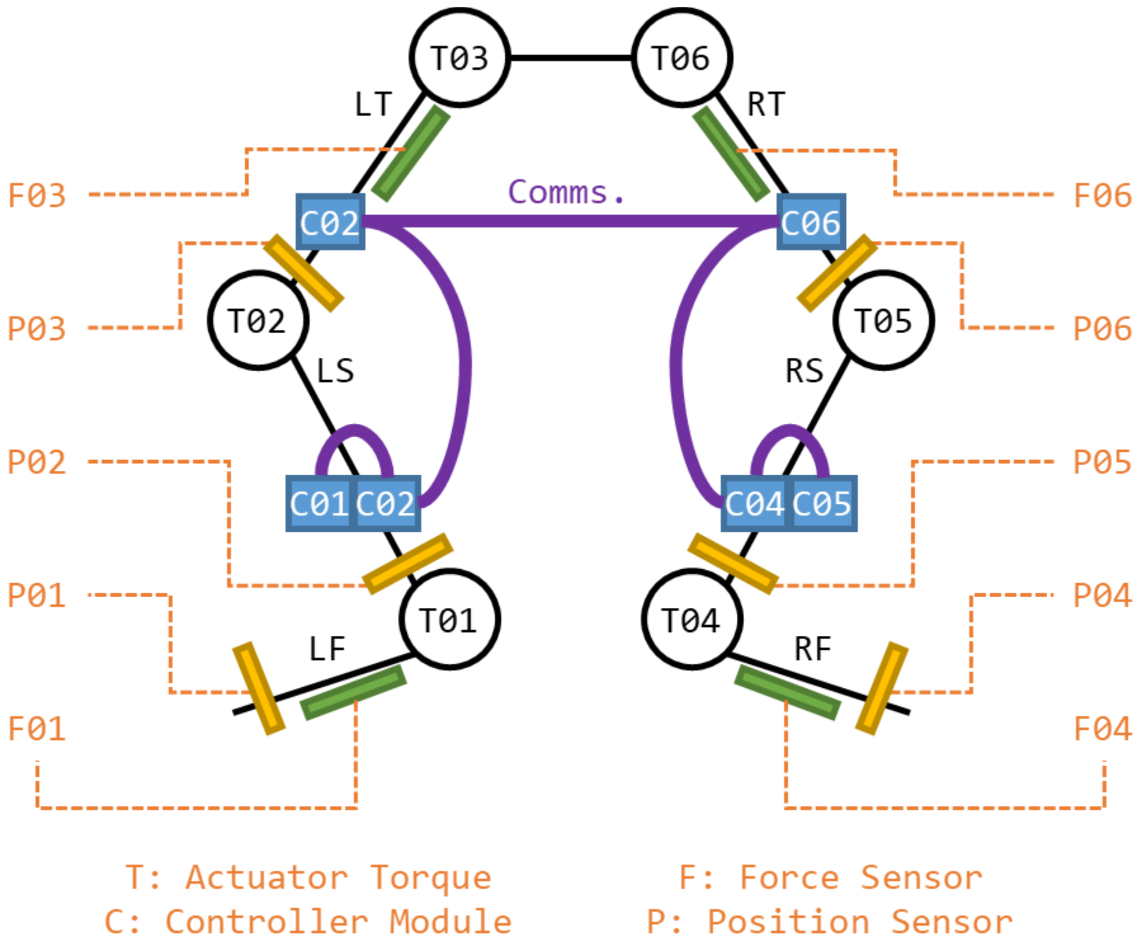


Figure 11: Exoskeleton Design

3.3 - Signal Flow Diagram - Figure 12 is a signal flow diagram detailing the system configuration. The configuration designed featured:

- 6 Position sensors at the end of each link;
- 4 force sensors, one at each of the contact zones;
- Control modules at the end[21] of each link; and,
- Daisy chain communications between the control modules.

[20] See **Error! Reference source not found. - Error! Reference source not found.**
[21] To reduce signal over cables loss from the position and proximity sensors by minimising cable distance.

For the controls and communication system to feature a consistent syntax regarding the system variables, each link and it associated sensors designated an ID, see Table 3.

Table 3: System Designations

Link	Designation	ID	Controller	Position	Force
Total System	**-**	**-**	**6**	**6**	**4**
Left Foot	LF	01	C01	P01	F01
Left Shin	LS	02	C02	P02	-
Left Thigh	LT	03	C03	P03	F03
Right Foot	RF	04	C04	P04	F04
Right Shin	RS	05	C05	P05	-
Right Thigh	RT	06	C06	P06	F06

To simplify the design process, modularise the controller design, and reduce the distance signals were to travel, a controller would be created for each limb section. Each controller would be physically identical, with the controllers for the shins[22] not interfacing with a force sensor. To reduce the load on the feet sections, C01 and C03 were located on the ankle.

As seen in Table 4, each controller was responsible for reading up to 6 values and communicating them to the other controllers. From this system configuration the requirements for the controllers could be defined.

Table 4: ADC Requirements

Component	IO
Controller	6 Inputs
Position Sensor	4 Outputs
Force Sensor	2 Outputs

3.2.3 Controller

The requirements for the controller to be selected were:

- Minimum of six ADC channels[23];
- Minimum of two UART channels;

[22] The controllers C02 and C05.

[23] An Analogue to Digital Converter (ADC) is a system that converts an analogue signal to a digital signal (Horowitz & Hill, 2015).

    - With DMA.
- Minimum of two PWM channels;
- 3.3 – 5V operating voltage (preferable);
- Integrated probe/debugger (preferable);
- USB programming (preferable).

The controller select was the STM32F303K8[24] on a STM NUCLEO-F303K8[25] development board. The STM32F303K8 met the requirements of the system:

- Ten ADC channels (eight after GPIO were allocated);
- Two UART channels;
    - With DMA.
- Fourteen PWM channels;
- 5V operating voltage[26];
- Integrated debugger/probe/debugger (in the form of a ST-LINK[27] on the development board);
- Supports the STM32CubeMX [28] initialization code generator; and,
- USB programming.

3.2.3.1 Code

Using STM32CubeMX the hardware and peripherals for the MCU[29] were enabled, the pinout for this configuration may be seen in 3.5 - STM32F303K8 Pinout - Figure 14. The settings for the peripherals may be found in their respective sections [30,31,32]. The code written to implement the system may be found in the attached code "\Code (C)" or 18 - Appendix H: Code Snippets. 3.4 - Process Flow Diagram - Figure 13 is process flow diagram detailing the systems behaviour.

[24] The STM32F303K8 datasheet may be found in the attached documents as "STM32F303x6x8 - Arm Cortex-M4 32b MCU+FPU [en.DM00092070].pdf".
[25] The STM NUCLEO-F303K8 STM NUCLEO-F303K8 datasheet may be found in the attached documents as "UM1956 - STM32 Nucleo-32 board [en.DM00231744].pdf".
[26] External or via micro USB
[27] The ST-LINK datasheet may be found in the attached documents as "TN1235 - Overview of the ST-LINK [en.DM00290229].pdf".
[28] The STM32CubeMX datasheet may be found in the attached documents as "UM1718 - STM32CubeMX for STM32 [en.DM00104712].pdf".
[29] Firmware on the STM32F303K8 must updated to the latest version for the system to boot correctly. The default firmware installed will not perform as designed.
[30] UART: see 7.2.1 - Communication Protocol
[31] PWM: see 8.2.4 - PWM Control
[32] ADC: see 3.2.5 - Reading Sensor Signals

3.2.4 Controller PCB

The controller MCU/development board was mounted on a dedicated controller PCB. The controller PCB would support:

- Headers for sensor connections[33], debugging[34], servo control, and system communication[35]
- Socket for the MCU;
- Power supply provisions;
  - Discussed in 3.2.8 - Power Supply;
- Axillary components (e.g. amplifiers for sensors); and,
- Indicator LEDs[36].

The schematics of the PCB created may be found in "\PCB Schematics\Controller" or the attached documents. The commissioned PCB may be seen in kt.

3.2.5 Reading Sensor Signals

Each board controller was to read up to six signal values. Readings were taken by utilising 6 of the 12 Bit ADCs[37] on the NUCLEO boards. These peripherals were initialised using STM32CubeMX with the settings shown in Table 5. The pins used, and their designated roles and IDs may be found in Table 6.

Table 5: ADC Configuration

Parameter	Value
ADCs_Common_Settings Mode	Independent mode
Clock Prescaler	ADC Asynchronous clock mode
Resolution	ADC 12-bit resolution
Enable Regular Conversions	Enable
Sampling Time	1.5 Cycles

[33] Fiction lock headers were used to prevent accidental removal.
[34] Specifically, a breakout for a UART channel.
[35] RJ45 jacks were used, as discussed in 7.2.3 - Connection Method
[36] Resistive dividers were used as step-down level shifters, variations in voltage and current were within acceptable parameters
[37] An Analogue to Digital Converter (ADC) is a system that converts an analogue signal to a digital signal (Horowitz & Hill, 2015).

Table 6: ADC Pins

ADC	Role	ADC	Channel
A	Force (Internal)	1	1
B	Force (External)	1	2
C	Position 1a	2	1
D	Position 1b	2	2
E	Position 2a	2	3
F	Position 2b	2	4

To update the ADC values a handler function was created[38] that would update the globally stored values for each ADC reading.

3.2.6 Intersystem communication

As the measurement of the system was to be conducted across multiple controllers a method of communicating within the system was required. Fortunately, the existing communication system was sufficient to enable intra and inter system communications. All controllers were connected in a daisy chain (C01 to C02 to C03 to C04 to C05 to C06) and would pass messages received on to the next board. This process is detailed in SS4[39].

3.2.7 Controller Mount

To affix the controller PCB in place, a mount was constructed that could be attached to the mount structure discussed in 19.1 - Mount Structure. The CAD for the mount is found in "\CAD Schematics" or the attached documents. For the mounting of controllers C01 & C02 and C03 & C04 is modified mount structure was created that could support two controller boards.

3.2.8 Power Supply

The NUCLEO boards could be powered via USB or an external supply[40]. When powered via USB the NUCLEO generated a 5V and a 3.3V rail. To power the controller board, and connected sensors boards, power was provided via the NUCLEO microUSB port. MicroUSB cables could then be connected to one of the two options created:

- A tethered solution: the microUSB to USB cables could be connected to a multiway USB charger connected to an extension cord allowing the system to be power via mains.

[38] See attached code "\Code (C)" or 18 - Update_ADC_Values
[39] See 7 - Approach and execution: SS4
[40] See 3.2.3 - Controller

- A portable solution: the microUSB to USB cables could be connected to 2000mAh (7.4kW) portable chargers.

This allowed for the batteries or power supply for the system to not be mounted on the legs of the exoskeleton, reducing torque requirements.

CAT 5e cables were used to connect the NUCLEO boards[41]. The cables were designed to provide RX and TX wired for the UART communication systems, however, as CAT 5e features 8 cables, the remaining cables were used to provide a common ground and 5V rail to connected boards. This allowed the power consumption of the boards and the system to be distributed amounts the system. Additionally, if a board were to lose power, the other boards could be used to power it via their 5V supply[42], essentially providing back up power to the system.

[41] See 7.2.3 - Connection Method

[42] Drawing the current for the entire system through a single device is not recommended, the system is designed to allow redundancy in powering.

3.3 Signal Flow Diagram

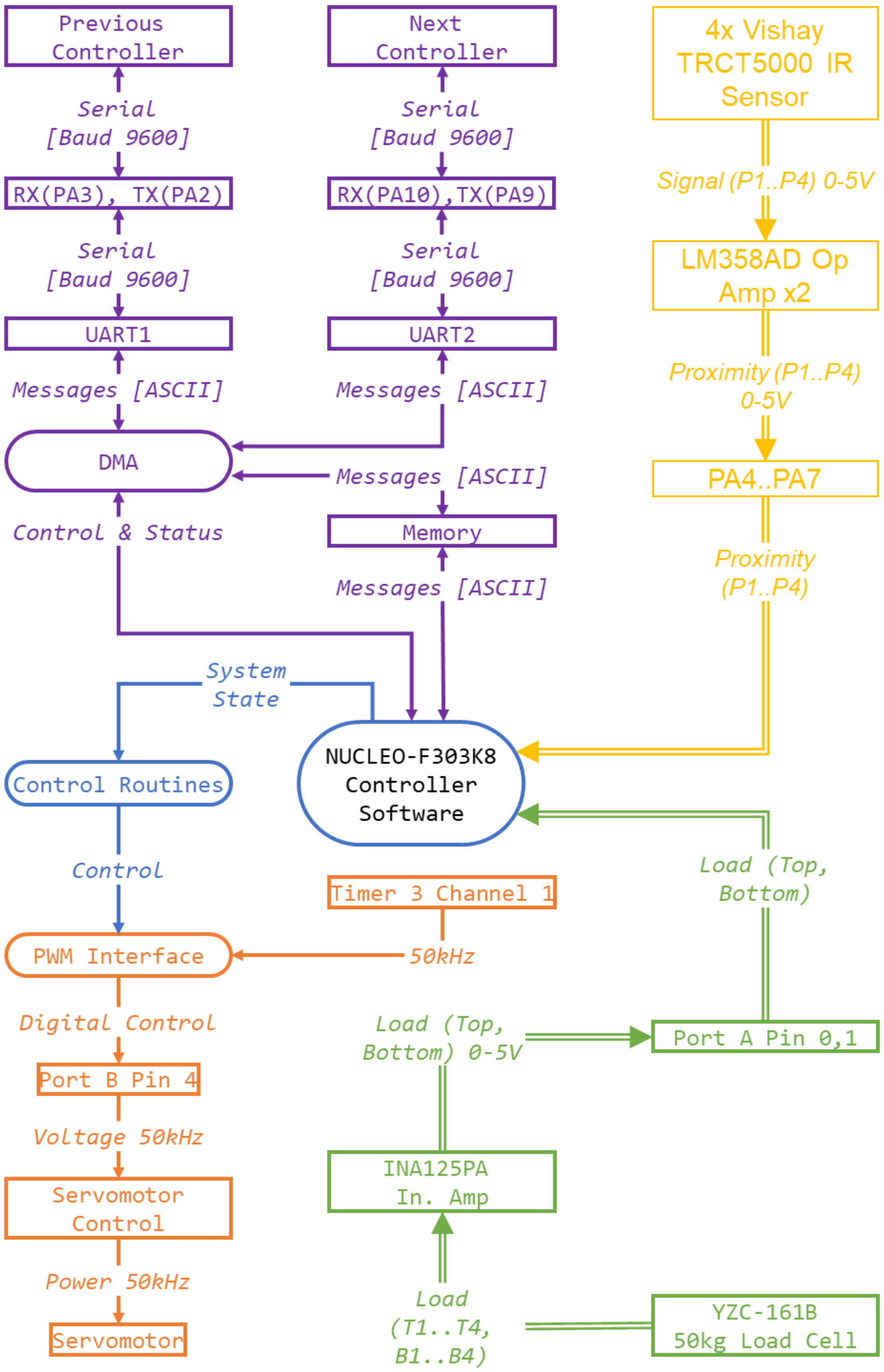


Figure 12: Signal Flow Diagram

3.4 Process Flow Diagram

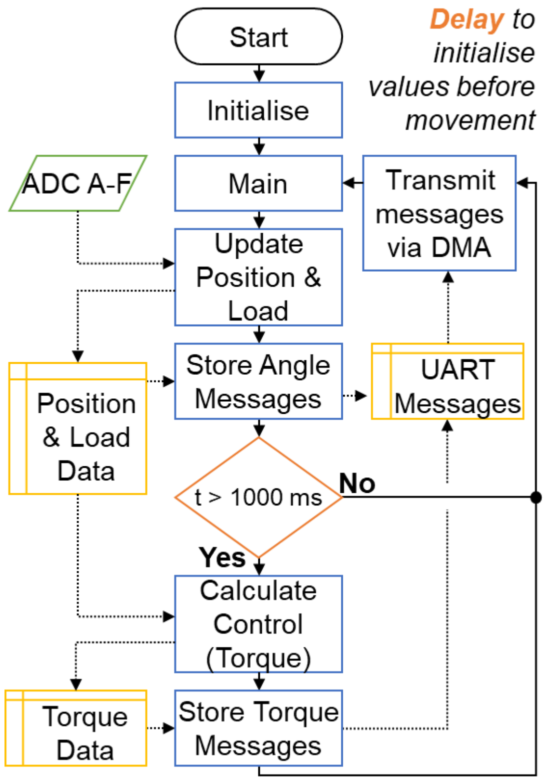


Figure 13: Process Flow Diagram

3.5 STM32F303K8 Pinout

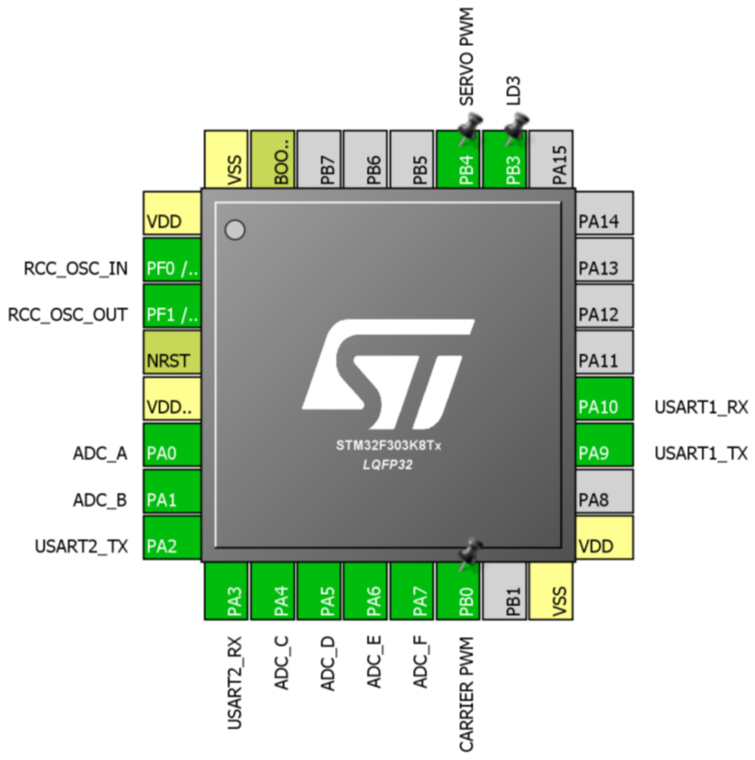


Figure 14: Controller Pinout

4. Approach and execution: SS1

4.1 Definition and Requirements

The purpose of subsystem one (SS1) was to detect the position of the pilot relative to the position of the exoskeleton in real time. This may be accomplished by measuring the position of limbs in relation to fixed rotational axis on suit.

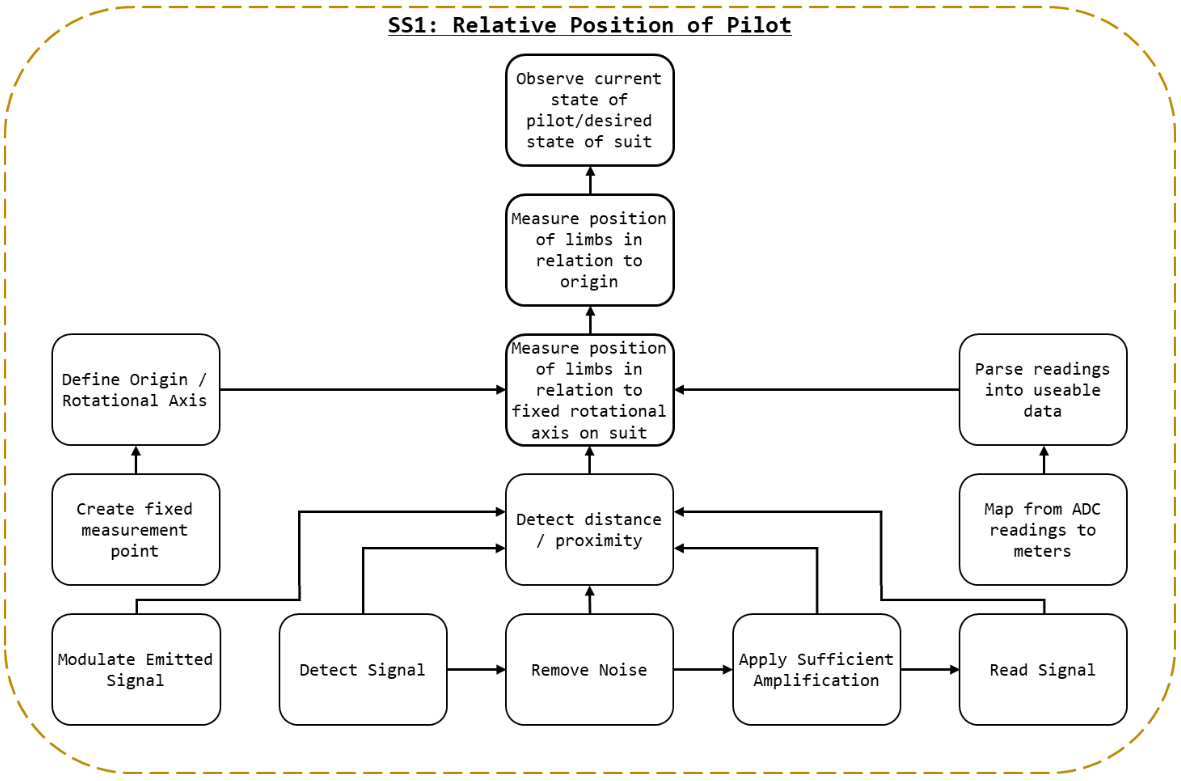

SS1: Relative Position of Pilot
Observe current state of pilot/desired state of suit
Measure position of limbs in relation to origin
Define Origin / Rotational Axis
Measure position of limbs in relation to fixed rotational axis on suit
Parse readings into useable data
Create fixed measurement point
Detect distance / proximity
Map from ADC readings to meters
Modulate Emitted Signal
Detect Signal
Remove Noise
Apply Sufficient Amplification
Read Signal

Figure 15: SS1 Breakdown

As detailed in Figure 15: SS1 Breakdown, to measure the position of limbs in relation to fixed rotational axis on suit:

- a fixed rotational axis must be defined[43];
- the position/distance must be measured; and,
- the measured distance must be parsed from raw values into useable data[44].

The process of measuring the distance will entailed:

- detecting a signal; the specific type depended on the techonlogy selected [45]; and,

[43] This implies a fixed point where readings can be taken, as such, a mechnism for fastening the detection system must be devised

[44] Functionally, this is the process of deriving the function that maps raw analogue voltage values to distance.

[45] E.g. IR light, ultrasonic waves, magnetic field strength, etc, see 13 - Appendix C: Proximity Sensors

- reading the signal, ostensibly with an ADC, into a format that could be parsed by the control systems.

4.2 Approach and Execution

4.2.1 Perceiving Distance

As seen in Figure 16, the position of a straight rod of a fixed length can be described by the relation of a point on the rod (as an angle) to a fixed rotational axis and origin. Therefore, to determine the current state of the pilot it is possible to observe the location of each link[46] in relation to a fixed rotation axis. By observing the position of the pilot's link in relation to the exoskeleton's, it is possible to know where the pilot is positioned[47].

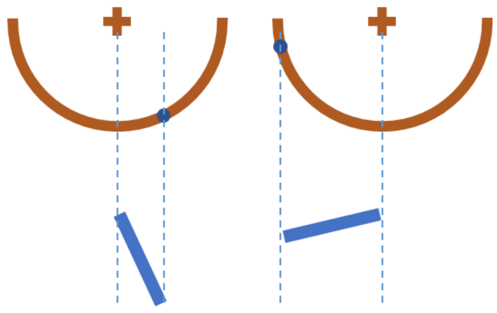


Figure 16: Point and Angle to Orientation

When considering the conditions of operation, the distance perception system was expected to take reading from amorphous human body parts, which may be clothed, shaved, hairy, firm, or soft. Body parts with rounded uneven surfaces at close ranges.

The conditions of operation featured many unknowns and the specific approach selected for the actuators could not be known prior to selection of the proximity sensing method. As such, the possibility of acoustic noise in the actuation system or the environment in general could not be dismissed.

Given this understanding of the operating conditions, ultrasonic sensors were considered inappropriate for the creation of a robust design within the constraints of the project. Instead IR range sensing was selected as the approach for determining distance as:

- The feature no minimum distance;
- While impeded function on uneven, rough, soft surfaces; and,

[46] Treated as a straight rod. This assumption isn't strictly necessary, if we can assume the shape of the link is known and the position of the pilot can be known. In further works, the mapping of physical coordinates and the controls may be updated to compensate for the true shape of human limbs.
[47] Assuming the position of the exoskeleton is known.

- Aren’t affected by acoustic noise or mechanical vibrations.

Additional sensors types considered found in 13 - Appendix C: Proximity Sensors.

The Vishay TCRT5000[48] was selected for the IR range sensing (Vishay Semiconductors, 2017). As stated by the manufacturer “The TCRT5000 and TCRT5000L are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light.” (Vishay Semiconductors, 2017).

A printed circuit board (PCB) would be created to which an IR transceiver, or emitter and receiver, could be mounted. The PCB was designed in Altium Designer (16.1), the PCB schematic may be found in the attached documents under the designation “IR Sensor Mount”. The PCBs were fabricated and assembled as shown in kt

4.2.2 Fixed Rotational Axis

The fixed rotational axis upon which distance measurements would be referenced was determined to be the hip, knee, and ankle joints, where ostensibly the actuators were to be place. As the size of links was dependent on the mechanical build[49] estimations where required for the majority of the project regarding the specific length of links. While it was presumed the exoskeleton’s, structural systems would be on the external sides of the pilot’s body, the mounts for the position detection system would be designed without specifies on materials or dimensions of the exoskeleton[50].

The mount structure design process may be found in 19.1 - Mount Structure.

Presuming a fixed mounting place (as given by the mount structure) a scaffolding structure would be required to mount the IR sensor PCBs in place. After testing the effective range of the IR sensors created it was determined that two sensors working in tandem provided the most reliable measurements of position. Two sensors doubled the effective IR emitted, and given the irregular surfaces of the human body, the readings could be averaged to give a more consistent reading.

The goals of the control systems was not to maintain an exact distance from the pilot, but instead to maintain the same distance on either side of the pilot. Rather than a solution comprised of measuring the pilot and attempting to maintain a specific offset, sensors on either

[48] The TCRT5000’s datasheet may be found in the attached documents as “TCRT5000 - Reflective Optical Sensor with Transistor Output.pdf”.
[49] By those responsible for the actuation system.
[50] These values would remain unconfirmed until exceedingly late within the project.

side of the pilot could be used to detect the difference in the offset on both sides. This way sensors would not need to be calibrated to the width of the pilot[51]

The design depicted in Figure 17: Sensor Frame was created. The frame (B) would envelope the pilot (A), and attach to the mount structure (F), see Figure 36: Mount Structure, on the outside edge of the pilot. A pair of sensors would detect in tandem (C) on either side the pilot (D), effectively measuring the position of the front and back of their leg in relation to the actuation of the exoskeleton.

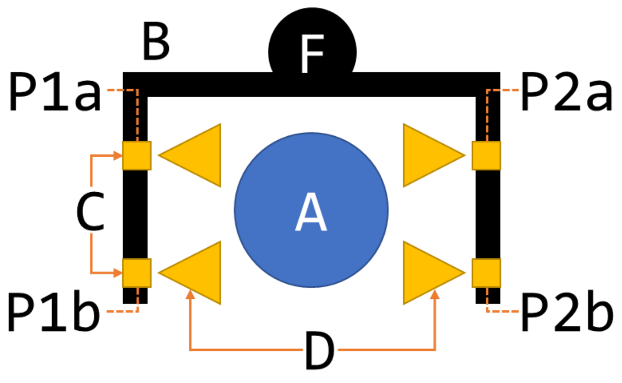


Figure 17: Sensor Frame

As shown kt, the sensor frame was constructed. Aluminium was selected for is strength and weight (relative to other metals). Once tested using the rig developed in 8.2.2 - Apparatus A: Position detection, a lighter material was sought. Wood was selected. While adequate for a prototype, the materials used should be replaced with lighter plastic or carbon fibre materials for future designs; the structural requirements of the system are minimal and weight reduction is a priority[52].

4.2.3 Mapping Values

As discussed in 1.3.1.2.1 - Variations, the lower extremity exoskeleton was never constructed. As such, the controls developed in 6 - Approach and execution: SS3 were not needed. Instead, as discussed in 6.2.7 - PID Tuning, an empirical method was used for the controls of the 1 DOF system. The result is that the explicit mapping from voltage to distance was not required, and only the difference in voltage mattered to the controls.

[51] Due to variations in bodily size.
[52] As noted in 3.1.1 - Additional Consideration

4.2.4 Amplification

Upon testing individual sensors, the system behaved correctly. However, when multiple sensors where tested at the same time on the same board they were found to have identical values at all times. This was inexplicable and previously unobserved behaviour.

When the microcontroller took ADC readings a capacitor was charged to hold a consistent voltage (the signal voltage) and the voltage of this capacitor was read. When reading directly from each of the sensors the capacitor would become charged, and when swapping to the next pin for the next reading rather than the sensor forcing the capacitor, the capacitor discharged into the sensor and drove the ADC voltage[53].

One solution was pausing to allow the capacitor to discharge, however this prevented the speed of measurement desired. Instead, a voltage follower[54] was implemented for each of the sensors so that the signal voltage would drive the ADC capacitor. For an op-amp the LM358AD[55] was selected.

In the PCB that was originally fabricated instead of pins 1-2 and 6-7 being shorted (for a gain of 1), pins 2-3 and 5-6 were shorted. To make a functional circuit, the physical traces on the fabricated PCB were severed. While unsightly. the solution worked, and readings taken from individual ADCs were correct. Future iterations of the PCB remedied the short.

[53] The MCU featured two ADC, each with their own capacitor, but each channel on the ADC shared a capacitor, so testing the two ADCs did not result in anomalous behaviour.
[54] See 16.1 - Voltage Follower
[55] The LM358AD's datasheet may be found in the attached documents as "LMV3xx - Low-Voltage Rail-to-Rail Output Operational Amplifiers.pdf".

5. Approach and execution: SS2

5.1 Definition and Requirements

The purpose of subsystem two (SS2) is to measure the force applied to the internals and externals of the suit.

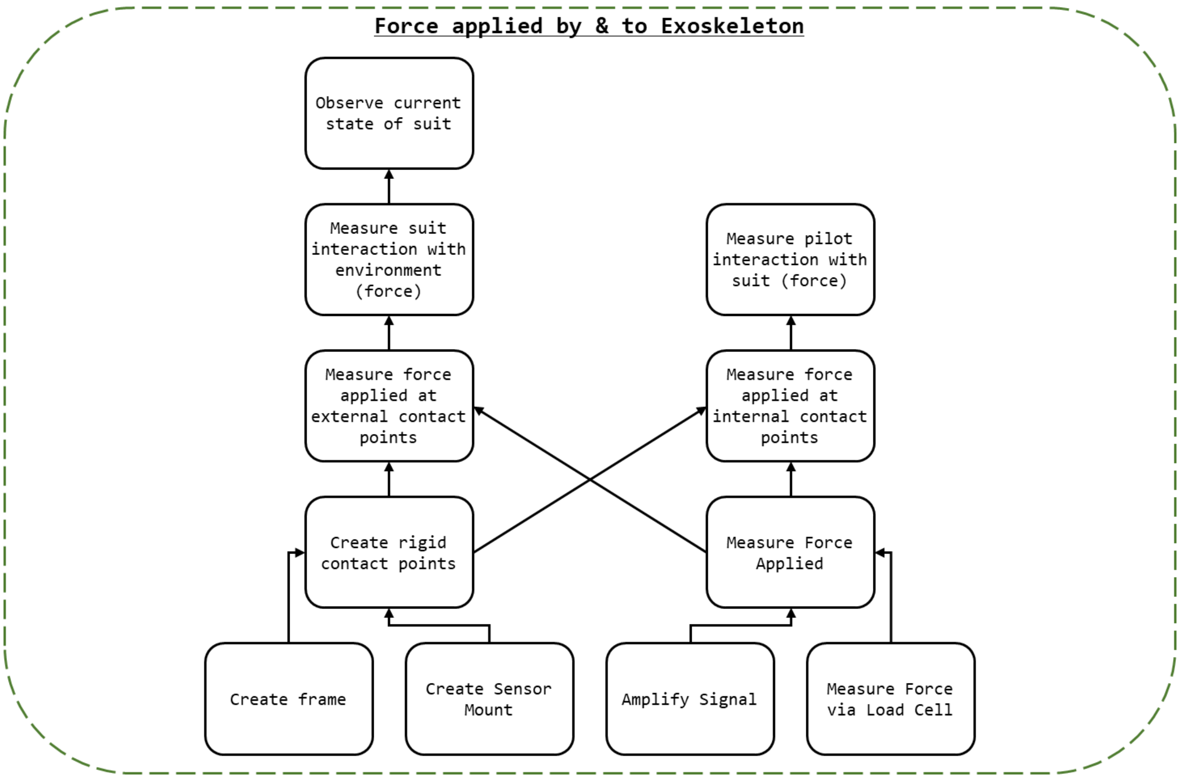

Force applied by & to Exoskeleton
Observe current state of suit
Measure suit interaction with environment (force)
Measure pilot interaction with suit (force)
Measure force applied at external contact points
Measure force applied at internal contact points
Create rigid contact points
Measure Force Applied
Create frame
Create Sensor Mount
Amplify Signal
Measure Force via Load Cell

Figure 18: SS2 Breakdown

As seen in Figure 18: SS2 Breakdown, to measure the force applied to the suit at the designated contact points (feet and upper thigh) the following was required:

- Creating of rigid contact point upon which force application could be measured[56];
- Measuring the force applied to the surface[57]; and,
- The measured distance must be parsed from raw values into useable data[58].

As stated in 1.3 - Scope, the pilot may apply force to the internals of the exoskeleton to indicate the desire to increase the force output of the suit. As a result, the force applied to the internals of the exoskeleton must be measured. For the representative motions required for a

[56] This would require a rigid frame upon which sensors could be mounted and Mounts for force sensors.
[57] This would involve measuring the force applied via load cell and, amplifying the signal from the load cells.
[58] Functionally, this is the process of deriving the function that maps raw analogue voltage values to force.

proof of concept; contact is only required with the ground and a seat, therefore the only locations where force regulation is required are the rear and the feet.

5.2 Approach and Execution

5.2.1 Measure Force

Load cells would be used to measure the force application to the contact points. Load cells allow for the precise measurement of force application in real time. Given the weight of the individuals associated with the project and preliminary estimations of the exoskeleton mass it was assumed that the mass for the entire system and pilot would not exceed 150 kg. Due to the incrementation of load cell ratings, four YZC-161B 50kg load cells[59] were selected. The YZC-161B[60] 50kg is a flat strain-gauge type load cell specifically rated for human mass measurement. The configuration of the YZC-161B may be found in Figure 19: YZC-161B Wire Configuration.

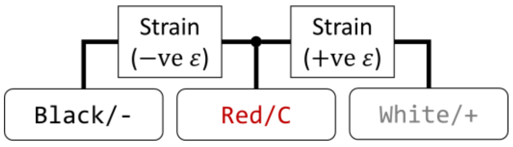
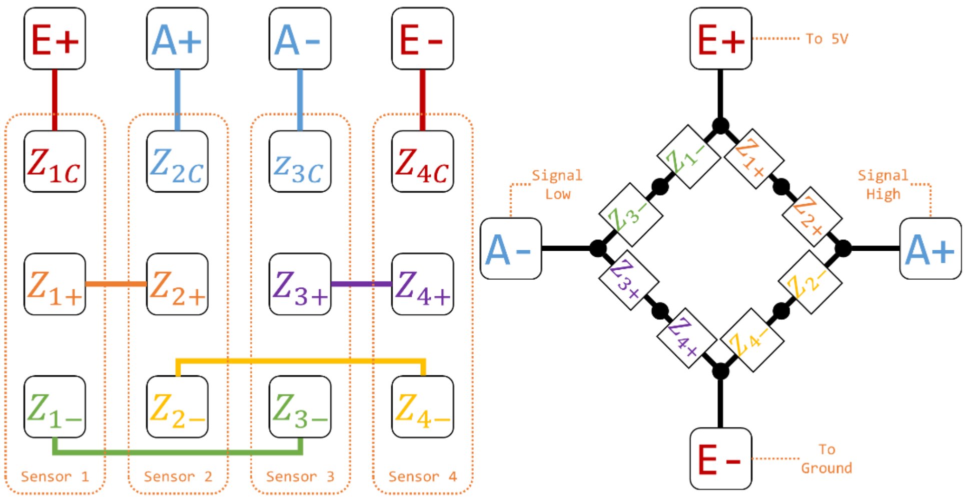


Figure 19: YZC-161B Wire Configuration

Four YZC-161B were wired in Wheatstone bridge configuration as shown in Figure 20: Load Cell Configuration[61].



[59] 200 kg rating for each contact point.
[60] The YZV-161B's datasheet may be found in the attached documents as "YZC-161B - 50kg Load Cell.pdf".
[61] Z refers to the impendence of the associate strain gauge, e.g. Z+1 = Positive stain gauge impendence for sensor one.

Figure 20: Load Cell Configuration

The configuration shown in Figure 20: Load Cell Configuration was implement in the PCB design of the controller boards, detailed in 3.2.4 - Controller PCB, in the schematic shown in Figure 21: Load Cell Topology, where B and T refer to the external and internal (bottom and top) load cell sets respectively.

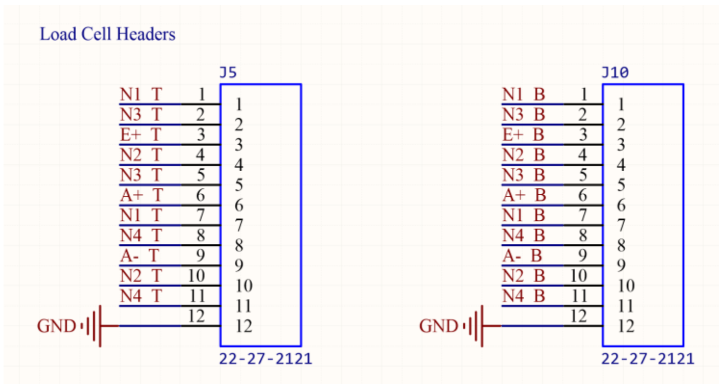


Figure 21: Load Cell Topology

5.2.2 Signal Amplification

To measure the signal output by the Wheatstone bridge configuration an instrumentation amplifier was used. The amplifier selected was the INA125[62]. The INA125s in the project were wired as depicted in Figure 22: Load Cell Amplifier Topology.

[62] The INA125's datasheet may be found in the attached documents as "INA125 - Instrumentation Amplifier With Precision Voltage Reference.pdf".

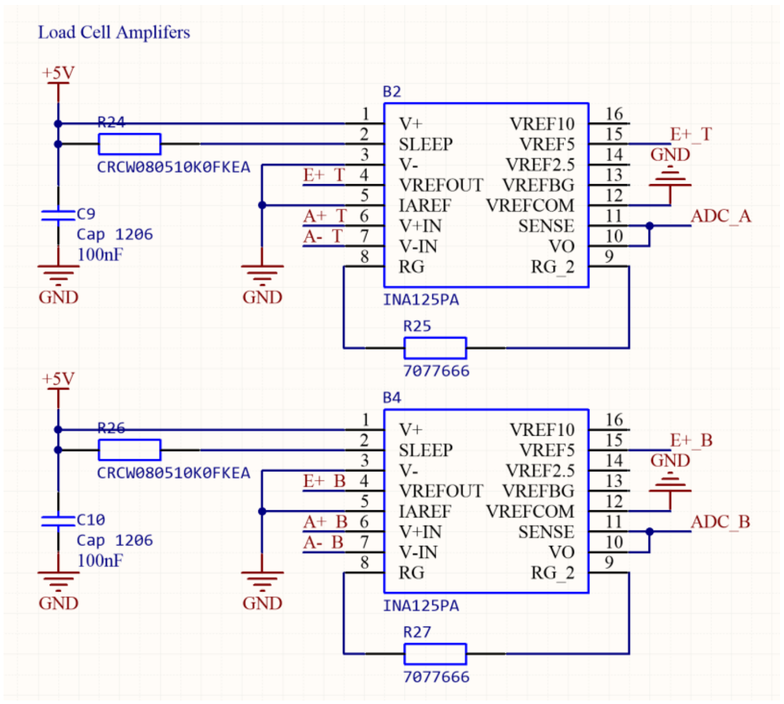


Figure 22: Load Cell Amplifier Topology

The INA125 allows for adjustable gain determined by the value of R_G, shown as R25 and R27 in Figure 22: Load Cell Amplifier Topology. The gain of the amplifier was given by Equation 7: INA125 Gain.

$$G = 4 + \frac{60\ k\Omega}{R_G}$$

Equation 7: INA125 Gain

Alas, the quality control for the YZC-161B was inconsistent[63] and the gain required could not be determined in advance[64]. Instead, socket headers were attached to a calibration board in place of resistors [65]. A load cell could then be attached to the board and calibrated, finding the resistor and gain that was most suitable.

As discussed in 1.3.1.2.1 - Variations, the demonstration to be conducted would not include a fully constructed exoskeleton and demonstration rigs were to be created. Consequentially, rather than configuring the load cells for full force range expected by the system, instead the

[63] This may have been the consequence of buying cheap Shenzhen parts.
[64] As seen in Table 7: Load Cell Calibration for two sensors set received in the same shipment, fresh from the packaging, the gain required for the same effective range of measurement was approximately ten times greater.
[65] So resistors could be added and removed without soldering.

load cells were configured so that the force applied by human hands would be sufficient to trigger a system response. To calibrate the load cells the following method was used:

1. Tare the load cells by recording the values taken at no weight applied;
2. Place a known weight on the load cells and record the readings;
3. Repeat step 2 with all available known weights; and,
4. Repeat steps 2 and 3 at least 3 times.

The results[66] were then processed, and the relationship between the sensor readings and mass were determined. As seen in Figure 23: Load Cell Calibration Data, the relationship between the voltage output and the mass is linear. Note that the change in resistance of the load cells is directly proportional, and the voltage follows the opposite relationship.

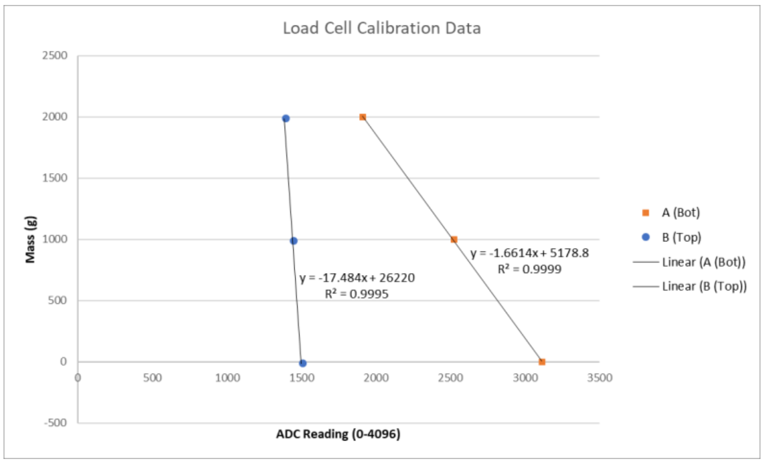


Figure 23: Load Cell Calibration Data

For the load cells used for the demonstration and test B, as detailed in 8.2.3 - Apparatus B: Force detection the configuration in Table 7: Load Cell Calibration was used.

Table 7: Load Cell Calibration

Load Cell Array	R_G	Gain	Voltage to Mass Function (as enacted in C)
Top (Internal)	4.7Ω	12800	massA = -1.6614 * ADC_A_Value + 5178.8;
Bottom (External)	47Ω	1280	massB = -17.484 * ADC_B_Value + 26220;

[66] Originally, water in a vessel was used to increase the weight from 1.2 kg to 6.9 kg in 0.25kg increments. However, the values were recalibrated after adjustments were made to the frame, using 0, 1, and 2 kg weights. The second set of results are depicted.

5.2.3 Rigid Frame

A rigid frame was required to ensure proper demarcation between internal and external force application. The frame was also to serve as the mount for the load cell mounts. Two aluminium plates where used with screw holes as needed. As noted in 5.2.2 - Signal Amplification, the load cells shipped did not match their description and feature prefabricated feet; rather than discarding them they were integrated into the design.

The design for the rigid frame with load cells and mounts may be found in Figure 24: Force Sensor Configuration. A rigid centre plate separated the internal and external load cells. A rigid top plate was added to the internal edge of the rig to protect the load cells and ensure an evenly distributed force from the pilot. In the case where the pilot applied force to the environment (A) force applied would be transmitted to the internal load cells, but the external load cells would remain independent (receiving force only from obstacles. In the case were obstacles applied force, but the pilot did not, the rigid plate would decouple the force allowing the controls system to indicate a stop condition.

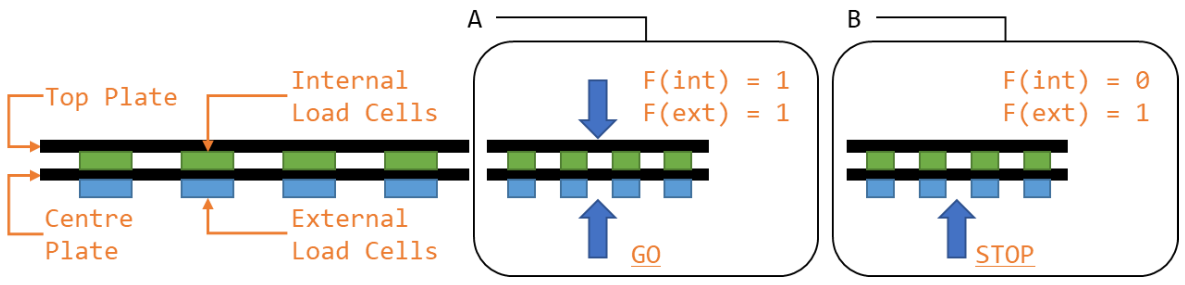


Figure 24: Force Sensor Configuration

The constructed force measurement system may be seen in kt

6. Approach and execution: SS3

6.1 Definition and Requirements

The purpose of subsystem three (SS3) was to determine the action that should be taken by the actuator system to minimise the error in the system.

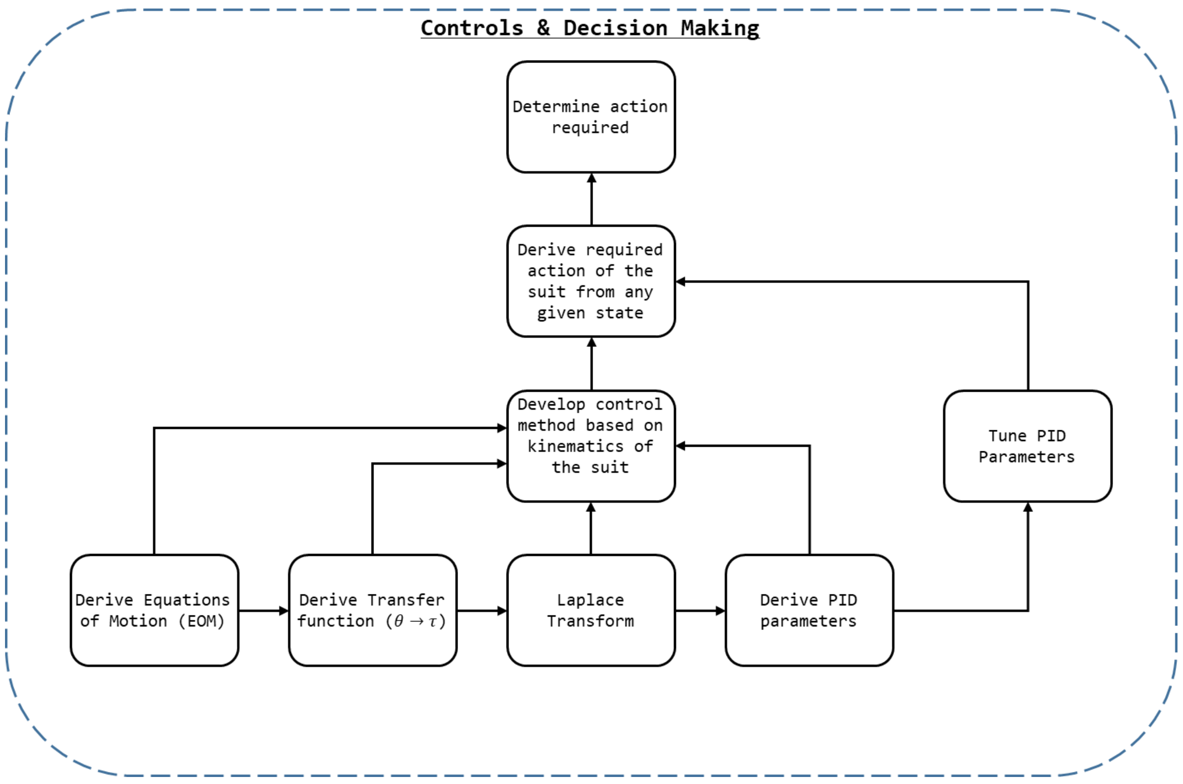

Controls & Decision Making
Determine action required
Derive required action of the suit from any given state
Develop control method based on kinematics of the suit
Tune PID Parameters
Derive Equations of Motion (EOM)
Derive Transfer function ($\theta \rightarrow \tau$)
Laplace Transform
Derive PID parameters

Figure 25: SS3 Breakdown

As seen in Figure 25: SS3 Breakdown, to determine the action required in a given state[67] the action required in any state must be known. To determine the general solution of what actions should be taken at any given state, the controls parameters and method for the system where be derived, and then this model was be refined by practically tuning the solution.

To determine the control parameters for the system the following method is employed:

1. Derive the equations of motion (EOM) for the system;
2. From the EOM derive the transfer function (TF) of the system (torque (τ) with respect to angle (θ));
3. Transform the TF into the Laplace Domain; and,
4. Derive the PID parameters from the Laplace Domain TF.

[67] Given by the values determined by SS1, and SS2

6.2 Approach and Execution

A 3 DOF control system was to control the lower extremity exoskeleton being constructed by the actuators and structural side of the project. Before the kinematics of the system could be found, the exoskeleton needed to be abstracted into a model. Consider the following:

a) The exoskeleton is to be affixed to the lower torso of the pilot;
b) The pilot is presumed to maintain the balance of the system using their body and should be able to manipulate the legs of the system independently; and,
c) The legs, while part of the same exoskeletons, are essentially fixed at the pelvis and operate independently and, we may therefore consider each leg as an independent manipulator with a fixed reference frame at the pelvis.

As noted in 1.3.1.3 - Exclusions (Out of Scope), each joint of the exoskeleton shall be constrained to 1 DOF. Therefore, we may abstract the exoskeleton as two 3 DOF RRR manipulators, as seen in Figure 26: Exoskeleton Abstraction.

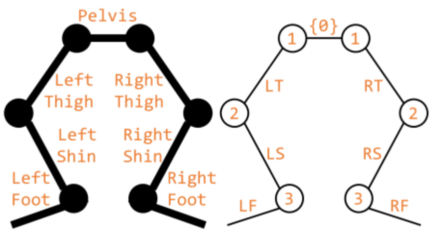


Figure 26: Exoskeleton Abstraction

For modelling the system, the parameters seen in Figure 27: 3 DOF RRR Parameterisation shall be used. Note angle shall be measured relative to the previous link with a clockwise positive convention.

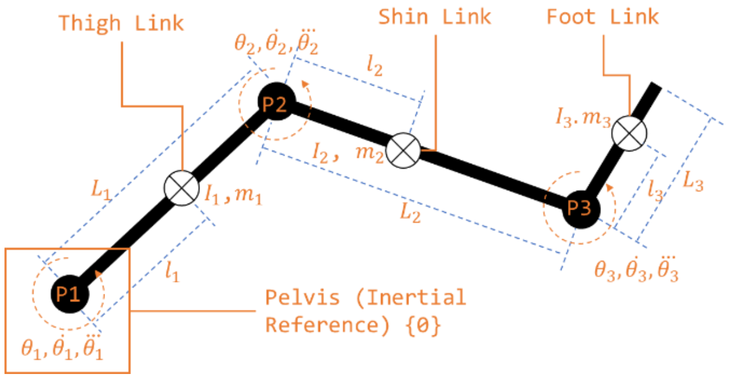

Thigh Link
Shin Link
Foot Link
$\theta_2, \dot{\theta}_2, \ddot{\theta}_2$
l_2
P2
I_3, m_3
I_2, m_2
L_1
I_1, m_1
L_3
l_3
L_2
P3
l_1
P1
$\theta_3, \dot{\theta}_3, \ddot{\theta}_3$
Pelvis (Inertial Reference) {0}
$\theta_1, \dot{\theta}_1, \ddot{\theta}_1$

Figure 27: 3 DOF RRR Parameterisation

As noted in 1.3.1.2.1 - Variations the actual values for the exoskeleton were never confirmed and the controls had to be completed symbolically. A general solution[68] was derived but the equations became cumbersome and large. As a result, many of the equations shall be taken to their general form[69], and explicit solutions shall be left as an exercise to the reader.

6.2.1 Jacobian

To find the Jacobian of a 3 DOF Revolute Manipulator for a system at low velocity [70]:

$$^{0}v_{P_1} = {}^{0}v_{P_0} + \omega_1 * P_1 = \begin{bmatrix} -l_1 s_1 & 0 & 0 \\ l_1 c_1 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

$$^{0}v_{P_2} = {}^{0}v_{P_1} + \omega_2 * P_2 = \begin{bmatrix} -(l_1 s_1 + l_2 s_{12}) & -l_2 s_{12} & 0 \\ l_1 c_1 + l_2 c_{12} & -l_2 c_{12} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

$$^{0}v_{P_3} = {}^{0}v_{P_2} + \omega_3 * P_3 = \begin{bmatrix} -(l_1 s_1 + l_2 s_{12} + l_3 s_{123}) & -(l_2 s_{12} + l_3 s_{123}) & -l_3 s_{123} \\ l_1 c_1 + l_2 c_{12} + l_3 c_{123} & -l_2 c_{12} + l_3 c_{123} & l_3 c_{123} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

The angular velocities are simply additive:

$$^{0}\omega_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \ \& \ {}^{0}\omega_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \ \& \ {}^{0}\omega_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

From which we obtain the Jacobian of a 3 DOF Revolute Manipulator, as seen in Equation 8.

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = J \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{pmatrix}$$

Equation 8: 3 DOF Revolute Manipulator Jacobian

[68] The solution found algebraically and implemented in MATLAB will work for any 1,2 or 3 DOF serial manipulator.

[69] 15 - Appendix E: Equations of Motion details the process of determining the explicit form of the equations of motion

[70] ${}^{0}v_{P_0} = 0$, Where movement of the total system is negligible in relation to rotational forces (i.e. standing, squatting, sitting, stairs, walking). At higher speeds the velocity of the pilot $\left({}^{0}v_{P_0}\right)$ must be considered.

6.2.2 Dynamics

For a 3 DOF Revolute Manipulator where the inertia tensors of the links are I_1, I_2, and I_3 (Equation 9).

$$^1I_1 = \begin{pmatrix} I_{xx1} & 0 & 0 \\ 0 & I_{yy1} & 0 \\ 0 & 0 & I_{zz1} \end{pmatrix} \& \ ^2I_2 = \begin{pmatrix} I_{xx2} & 0 & 0 \\ 0 & I_{yy2} & 0 \\ 0 & 0 & I_{zz2} \end{pmatrix} \& \ ^3I_3 = \begin{pmatrix} I_{xx3} & 0 & 0 \\ 0 & I_{yy3} & 0 \\ 0 & 0 & I_{zz3} \end{pmatrix}$$

Equation 9: Inertia Tensors

6.2.2.1 Explicit form of Manipulator Mass Matrix

The mass matrix for the 3 DOF Revolute Manipulator is given by Equation 10[71].

$$M = m_1 J_{v1}^T J_{v1} + J_{\omega 1}^T I_1 J_{\omega 1} + m_2 J_{v2}^T J_{v2} + J_{\omega 2}^T I_2 J_{\omega 2} + m_3 J_{v3}^T J_{v3} + J_{\omega 3}^T I_3 J_{\omega 3}$$

Equation 10: Mass Matrix for the 3 DOF Revolute Manipulator

6.2.2.2 Vector of centrifugal and Coriolis forces

We begin with Equation 33[72].

$$C(q) = \begin{bmatrix} b_{111} & b_{122} & b_{133} \\ b_{211} & b_{222} & b_{233} \\ b_{311} & b_{322} & b_{333} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & \frac{1}{2}(m_{122} + m_{122} - m_{221}) & \frac{1}{2}(m_{133} + m_{133} - m_{331}) \\ \frac{1}{2}(m_{211} + m_{211} - m_{112}) & 0 & \frac{1}{2}(m_{233} + m_{233} - m_{332}) \\ \frac{1}{2}(m_{311} + m_{311} - m_{113}) & \frac{1}{2}(m_{322} + m_{322} - m_{223}) & 0 \end{bmatrix}$$

Equation 11: Vector of Centrifugal Forces

Next Equation 34[73].

[71] This process was completed with symbolic variables in MATLAB R2017b, as detailed in the attached files "\Code (Matlab)\get_EOM.pdf"
[72] This process was completed with symbolic variables in MATLAB R2017b, as detailed in the attached files "\Code (Matlab)\get_EOM.pdf"
[73] This process was completed with symbolic variables in MATLAB R2017b, as detailed in the attached files "\Code (Matlab)\get_EOM.pdf"

$$B(q) = \begin{bmatrix} 2b_{112} & 2b_{113} & 2b_{123} \\ 2b_{212} & 2b_{213} & 2b_{223} \\ 2b_{312} & 2b_{313} & 2b_{323} \end{bmatrix}$$
$$= \begin{bmatrix} m_{112} & m_{113} & (m_{123} + m_{132} - m_{231}) \\ 0 & (m_{213} + m_{231} - m_{132}) & m_{223} \\ (m_{312} + m_{321} - m_{123}) & 0 & 0 \end{bmatrix}$$

Equation 12: Vector of Coriolis Force

6.2.2.3 Vector of gravity force

Continuing from Equation 37: Vector of Gravity Force

$$G = -(J_{v1}^T \quad J_{v2}^T \quad J_{v3}^T)\begin{pmatrix} m_1 g \\ m_2 g \\ m_3 g \end{pmatrix} = -g(J_{v1}^T m_1 + J_{v2}^T m_2 + J_{v3}^T m_3)$$

Equation 13: Gravity Vector

Given J_{vi}^T [74], we may find Equation 14[75].

$$G = \begin{bmatrix} \big((m_1 + m_2 + m_3)l_1 c_1 + (m_2 + m_3)l_2 c_{12} + m_3 l_3 c_{123}\big)g \\ \big((m_2 + m_3)l_2 c_{12} + m_3 l_3 c_{123}\big)g \\ (m_3 l_3 c_{123})g \end{bmatrix}$$

Equation 14: Vector of Gravity Force

6.2.3 Explicit Form of the Equations of Motion

Base on Equation 10, Equation 11, Equation 12, and Equation 14 we may find the solution to Equation 24, as given by Equation 15[76]

$$M(q)\ddot{q} + C(q)[\dot{q}^2] + B(q)[\dot{q}\dot{q}] + g(q) = \tau$$

Equation 15: Equations of Motion

The output of these equations, and the equations of motion for the system are as seen in 15 - Appendix E: Equations of Motion - Table 15: Equations of Motion.

6.2.4 Laplace Transform

Once the equations of motion (EOM) had been found via MATLAB, the Laplace transform was performed on the EOM. Alas, MATLAB doesn't recognise the relationship between θ_i,

[74] See Equation 8

[75] This process was completed with symbolic variables in MATLAB R2017b, as detailed in the attached files "\Code (Matlab)\get_EOM.pdf"

[76] This process was completed with symbolic variables in MATLAB R2017b, as detailed in the attached files "\Code (Matlab)\get_EOM.pdf"

$\dot{\theta}_i$, and $\ddot{\theta}_i$ for *syms*; they had to be treated as separate variables. Thus, the Laplace transform could not be applied outright. Instead, it would be performed manually by transforming each term individually through string manipulation[77]. This can be seen in detail in attached files "\Code (Matlab)", and involved finding all instances of a variable (e.g. cos(a3)) and transforming it (e.g. 'A3*(s/(s^2 + 1))')[78].

While 3D Laplace transforms are possible (Dawkins, 2018) it unnecessary[79]. Instead it will be assumed the behaviour of the systems in relation to a joint can be approximated to equal the behaviour of the system if the instantaneous state of the rest of the joints is constant[80].

The resulting equations may be found in 15 - Appendix E: Equations of Motion - Table 16: Laplace EOM.

6.2.5 Transfer Functions

Given the solution found in Table 16: Laplace EOM, we may rearrange the equations found so the solution is given in terms of the transfer function $\left(TF = \frac{\theta_i}{\tau_i}\right)$. The transfer functions for the system may be found in 15 - Appendix E: Equations of Motion - Table 17.

6.2.6 PID Development

At this point we in many ways reach the limits of symbolic variables in MATLAB. PIDs can be created with tuneable parameters, so it is possible to treat the system's variables (mass, length, etc…) as tuneable values preserving our purely algebraic solution[81]. Instead it is more prudent to develop the software to find the PID parameters for the system once basic fundamentals regarding the system can be confirmed.

[77] MATLAB stores symbolic equations as strings.
[78] This of course had to happen from greatest term to smallest to avoid confusions in replacements (i.e. a3 includes dda3, da3, and a3, so it should be completed after all terms including dda3 and da3 are transformed).
[79] And messy.
[80] This is to say, if we update θ_i sufficiently quickly/frequently we may treat changes in θ_j and θ_k as negligible.
[81] However, this would exist as a purely intellectual exercise of no practical merit.

See "\Code (Matlab)\materialise.pdf" for MATLAB code which shall substitute all systems variables for their actual variables (when confirmed). This shall allow the reader to solve for the "actual" PID values for their system[82][83].

6.2.7 PID Tuning

The servomotors used in testing featured their own embedded control systems, and their torque, angle, velocity, and acceleration could not be directly controlled. As such precise controls could not be derive from first principles. Instead the controls systems were tuned empirically to achieve the desired system response.

Empirical PID tuning methods, like the second Ziegler-Nichols method, allow for the tuning of PID values where the exact mathematical model of the system to be controlled is unknown.

As such, the second Ziegler-Nichols method for determine PID values, and PID control, were selected as the control structure for the system. Following the second Ziegler-Nichols method the control parameters for the PID where empirically determined, as seen in Table 8, the PID control was then implemented in C, as seen in 18.8 - PID.

Table 8: 1 DOF PID Parameters

Control Parameter	Value
K_p	5000
K_i	500
K_d	0

[82] Angles should be entered as tuneable parameters, this can then be used to find the variation in PID parameters for the full range of movement. Analysis has found the variation sufficiently small for the range of human leg movement that PID parameters may be approximated as constant. This would allow for the creation of three independent SISO PID loops, opposed to a MIMO system.
[83] From this point it is trivial to plug the transfer functions into MATLAB's pidTuner and generate a PID, and as such will not be documented here. It does not provide actual values for control (these cannot be found until there is an exoskeleton to control), and the documentation for pidTuner is sufficient for self-direction.

The K_d value found equals zero, this is discussed further in 9.1.6 - SS3. The resulting system is not a PID, but a PI controller, as seen in Figure 28, with a transfer function as seen in Equation 16.

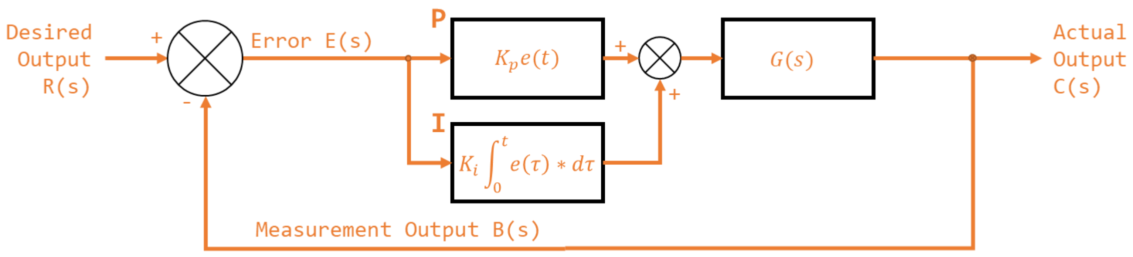


Figure 28: PI Control Block Diagram

$$K(s) = K_p + \frac{K_i}{s}$$

$$H(s) = G(s)K(s) = G(s)(K_p + \frac{K_i}{s})$$

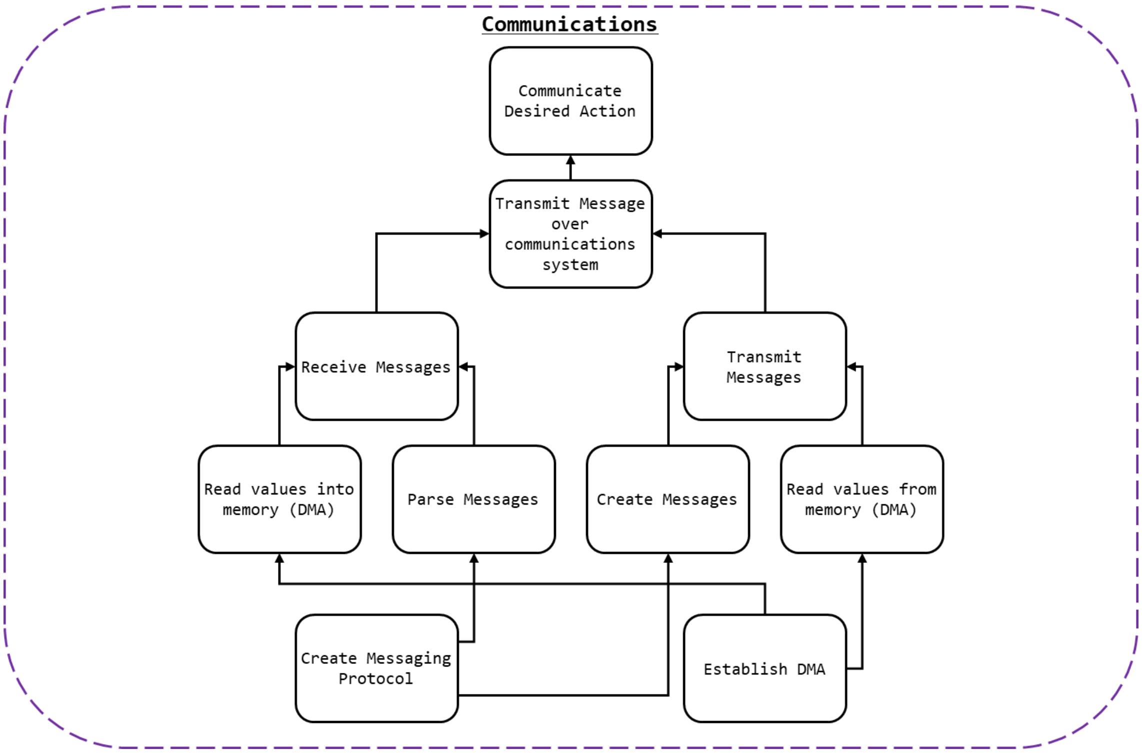
$$TF = \frac{G(s)(K_p + \frac{K_i}{s})}{1 + G(s)(K_p + \frac{K_i}{s})} = \frac{G(s)(5000 + \frac{500}{s})}{1 + G(s)(5000 + \frac{500}{s})}$$

Equation 16: PI Transfer Function

7. Approach and execution: SS4

7.1 Definition and Requirements

The purpose of subsystem four (SS4) is to communicate the desired action to the actuation system.



Communications
Communicate Desired Action
Transmit Message over communications system
Receive Messages
Transmit Messages
Read values into memory (DMA)
Parse Messages
Create Messages
Read values from memory (DMA)
Create Messaging Protocol
Establish DMA

Figure 29: SS4 Breakdown

As detailed in Figure 29: SS4 Breakdown, to communicate the desired action to the actuation system messages would need to be received[84] and sent[85]. To ensure that messages where properly interpreted and created by all parties[86], a communication protocol and packet protocol would be need.

7.1.1 Communication Protocol

The requirements of the communication protocol where as follows:

- ASCII messages must be transmittable;
- Synchronisation or dependency between systems is to be avoided;
- Systems which required addresses for communication were to be excluded.

[84] This process would involve receiving the messages and parsing them into useable data
[85] This process should involve creating messages and transmitting them.
[86] Controls/Perception and Actuation/Structural subprojects.

7.1.2 Packet Protocol

The requirements of the packet protocol where as follows:

- Generic data types must be transmittable;
- Signed values[87] of either integers or real numbers would be sent; and,
- Start and stop bits characters would be used to indicate the beginning and the end of messages[88].

7.1.3 Connection Method

The requirements of the connection where as follows:

- Wired connection method;
- A common ground should be established (where relevant); and,
- Where possible standardised jacks/sockets should be used.

7.2 Approach and Execution

7.2.1 Communication Protocol

As noted in3.2.3 - Controller, two communication channels were required per microcontroller. Two UART channels where implemented in C for the STM32 NUCLEO boards. The pin used by each channel is shown in Table 9: UART Pins.

Table 9: UART Pins

UART	Function	STM32F303k8 Pin	NUCLEO Connector Pin	DMA Channel
UART 1	RX	Port A Pin 10	D0	5
	TX	Port A Pin 9	D1	4
UART 2	RX	Port A Pin 3	A2	7
	TX	Port A Pin 2	A7	6

Much of the peripheral initialisation was completed used STM32CubeMX. The configuration file used can be found in the attached documents or in the file structure[89]. The configuration of both UART channels may be found in Table 10: UART Configuration.

[87] Positive or negative numbers.
[88] In the case of incomplete messages, the message should be invalid.
[89] See "\Code (C)\STMcubeMX - Apparatus A - Position.pdf", "\Code (C)\Apparatus A - Position_Kamikaze_Phoenix_Kamikaze_Phoenix.ioc", "\Code (C)\STMcubeMX - Apparatus B - Force.pdf", or "\Code (C)\Apparatus B - Force_Lucky_Boar_Lucky_Boar.ioc"

Table 10: UART Configuration

Parameter	Value
Baud Rate	9600 Bits/s
Word Length	8 Bits (Including Parity)
Parity	None
Stop Bits	1
Alternate Function (for GPIO)	7

Each UART was configured to use DMA to receive and transmit messages. The DMA settings can be found in Table 11: DMA Configuration.

Table 11: DMA Configuration

Parameter	Value
DMA Mode	Normal (Not circular)
Data Width (Peripheral)	Byte
Data Width (Memory)	Byte
Increment Address	Memory
Priority	Medium

The process of transmitting a message was as follows:

- Ensure UART 1 and 2 were ready for transmission[90];

1.1 [90] See attached code "\Code (C)" or 18.8 - PID

```
/**
 * @brief    Updates the control value, this is the output of K(s)
 * and is our command to the actuators to achieve error == 0
 * @param    par : system paramaters
 */
void update_control(struct PARAMETERS* par) {

	// get e
	e = get_p_target(par) - get_p(par);

	// update moving average for intergral term
	maaPush(par->q, e);

	// Error for proportional term
	Ep = e;
	// Error for integral term
```

- Load message into buffer; and,
- Direct UART to transmit buffer via DMA[91].

7.2.2 Packet Protocol

As seen in Figure 30: Packet Protocol, a packet protocol was established to ensure that messages could be understood. The protocol is detailed in Table 12: Packet Protocol and was used for all communication within the controls/perception system.

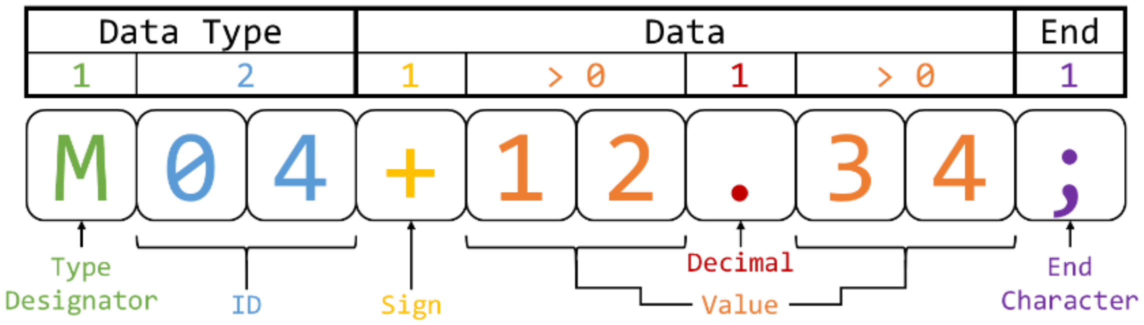


Figure 30: Packet Protocol

Table 12: Packet Protocol

Value	Notes
Type	The designator indicated what type of value was being transmitted[92].
ID	The integer representing which value of a given type was being updated[93].
Sign	Used to indicate if the value was positive or negative.
Value	The value being updated. A left-hand side and right-hand side value was required for a valid message[94].

```
        Ei = get_integral(par->q);
        // Error for derivatve term
        Ed = e - par->e_last;

        // Set the target torque
        set_T_target(par, (par->Kp * Ep + par->Ki * Ei + par->Kd * Ed));
        par->e_last = e;
        return;
}
```

Is Transmitting

[91]See "\Code (C)\...\Drivers\STM32F3xx_HAL_Driver" - HAL_UART_Transmit_DMA

[92] For example, 'T' (or 't') indicated that the value represented the desired torque of an actuator.

[93] For example, an ID of 03 referred to the right foot. See 3.2.2 - System Configuration

[94] E.g. 11 or .5 was invalid, but 11.0 and 0.5 was valid.

Decimal	Used to punctuate the value received[95].
End	Used to indicate the end of a message[96].

7.2.3 Connection Method

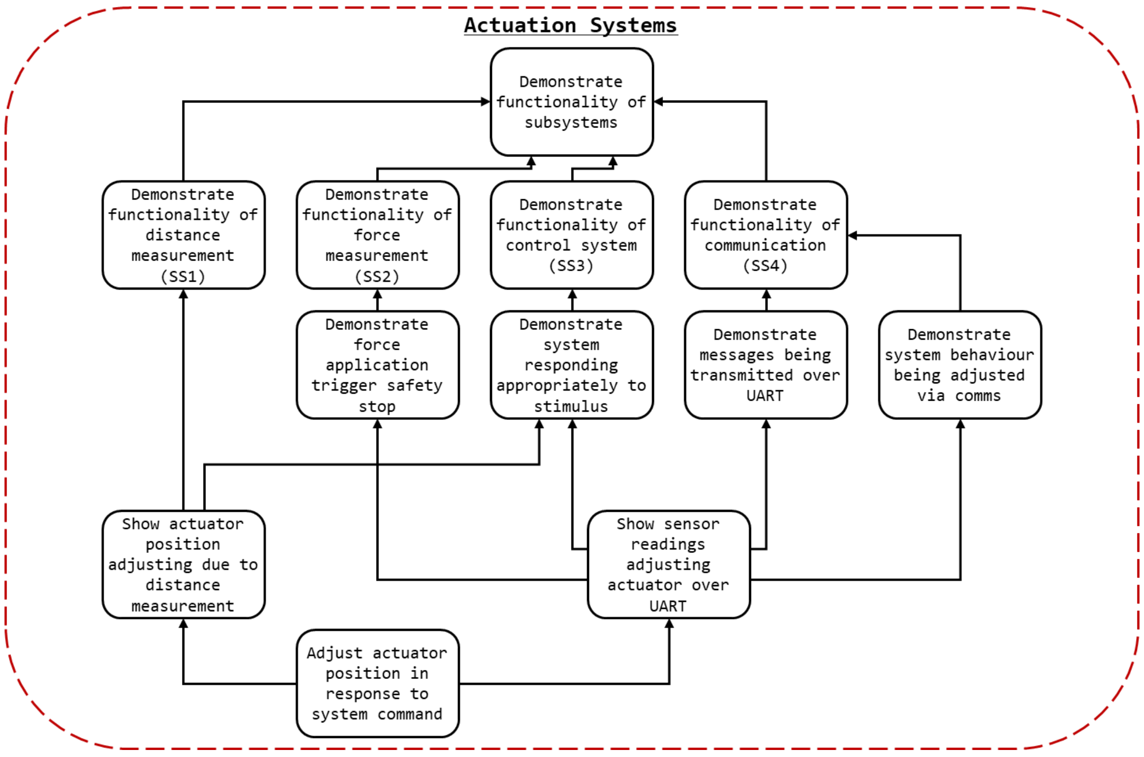
As UART was to be used over a two-wire connection three wires in total were required: RX, TX, and Ground[97]. While numerous cable options where viable, Category 5 cable[98], or CAT 5, cables where selected to connect devices together. CAT 5 cable were terminated with 8P8C modular connectors and plugged into female RJ45 connectors[99]. These connectors were then mounted on the controller PCB[100].

[95] Indicated the end of the left-hand side of the value.
[96] Upon reading a valid message this value would be changed to an '@' to prevent the same message (within the DMA buffer) from being read twice.
[97] A common ground ensures that all RX an TX messages had the same reference voltage and where read correctly.
[98] CAT 5 cables are twisted pair cables (4 sets) commonly used in ethernet connection
[99] As CAT 5 cables feature more than three wires it became possible to provide 5V across boards, this is discussed in further detail in 3.2.8 - Power Supply.
[100] See 3.2.4 - Controller PCB

8. Approach and execution: SS5

8.1 Definition and Requirements

The purpose of subsystem five (SS5), in lieu of the final exoskeleton actuator system, is to demonstrate the functionality of the other subsystems (and the project in general).



Actuation Systems
Demonstrate functionality of subsystems
Demonstrate functionality of distance measurement (SS1)
Demonstrate functionality of force measurement (SS2)
Demonstrate functionality of control system (SS3)
Demonstrate functionality of communication (SS4)
Demonstrate force application trigger safety stop
Demonstrate system responding appropriately to stimulus
Demonstrate messages being transmitted over UART
Demonstrate system behaviour being adjusted via comms
Show actuator position adjusting due to distance measurement
Show sensor readings adjusting actuator over UART
Adjust actuator position in response to system command

Figure 31: SS5 Breakdown

As detailed in Figure 31: SS5 Breakdown to demonstrate the functionality of the other major subsystems two representative action sets can be completed[101]. The functionality of SS2, SS3, and SS4 can be demonstrated by showing that force sensors readings from one device, over the communication system, can shut off the operation of an actuator when the external force exceeds the internal force. The functionality of SS1 and 3 can be shown by demonstrating that the position of an actuator can be controlled by position sensor readings. The functionality of both these tests however, is dependent on the behaviour of an actuator as directed by the system.

[101] The reason for two tests, rather than a single integrated test was a consequence of servomotor lead time, as noted in 8.2.1 - Actuators.

8.2 Approach and Execution

8.2.1 Actuators

Due to the time constraints associated with creating SS5 and the goal of demonstrating SS1-4[102], rather than creating a servomotor a prefabricated servomotor was selected as the desired actuator for SS5.

As the lead time associated with most high precision servomotors was beyond the remainder of the project when SS5 was commissioned[103] the selection of servomotors available to the project was limited. As such, there were no servomotors available capable of actuating with the force sensor plate attached[104].

For demonstrating the efficacy of the position system, greater torque would allow for greater control authority when accelerating, and as the mock exoskeleton[105] would be constructed from suboptimal materials greater control authority was a priority. Thus, the MG995[106] servomotor was selected. The MG995 offered position-based control and 0.98 Nm (10 kgf cm) of torque (at 6V)[107].

For demonstrating the efficacy of the force detection systems, a continuous rotation servo was selected as it could stop when the external force on the suit exceeded the force internal and could vary its speed depending on the force applied internally. Due to its availability the 900-00008[108] Continuous Rotation Servomotor was selected[109].

To control the servomotors a PWM generated by the NUCLEO boards was used. As the PCBs designed[110] did not consider SS5 within scope at the time of fabrication, there were no dedicated headers for powering the servo. Later iterations remedied this by adding a dedicated servo header. For creating the configurations discussed in SS5 fly-wires were solder to the board, and the connections were heat shrunk,. While suboptimal it did allow for reliable and consistent control of the servomotors.

[102] Rather than creating a functional exoskeleton
[103] In response to learning there would be no exoskeleton.
[104] The torque requirements were too high
[105] See 8.2.2 - Apparatus A: Position detection
[106] The MG995 datasheet may be found in the attached documents as “MG995 - High Speed Metal Gear Dual Ball Bearing Servo.pdf”.
[107] The strongest servomotor available on short notice.
[108] The 900-00008 datasheet may be found in the attached documents as “900-00008 - Continuous Rotation Servo.pdf”.
[109] In truth, the 900-00008 was originally proposed for the position system but lacked the torque.
[110] See 3.2.4 - Controller PCB

8.2.2 Apparatus A: Position detection

The first apparatus, to demonstrate SS1 and SS3, would entail attaching the position detection system to a rod at the end of a servomotor and controlling the motor/rod position based on the position readings. Effectively a two-link/one-joint mock exoskeleton.

As seen in Figure 32: Apparatus A Configuration, A fixed exoskeleton link (A) is installed flush with a pilot link (B) (e.g. thigh, bicep). At the joint and actuator (S) is used to rotate a free exoskeleton link (C) flush and parallel to a link of the pilot (D) (e.g. shin, forearm). The IR sensor frame (E) is mounted to the end of C and IR sensors (F) may determine the position of D.

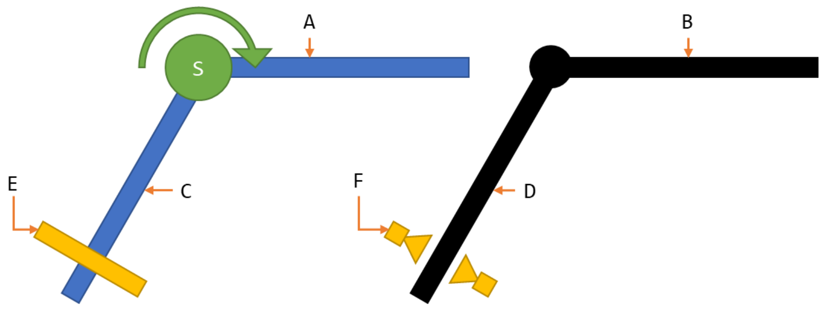


Figure 32: Apparatus A Configuration

The MG995 was used as the actuator for Apparatus A. Aluminium beams (x mm) kt were used for the exoskeleton links. To minimise the weight of the perception system, the power supply for the servo, SS1, and SS3 were located on the upper link. Additionally, the controls board was also located on the upper link.

To mount the lower link to the servomotor, cable ties were used to attach the segment to a servo horn. A bolt hole placed in the segment was then used to screw the servo horn, servomotor, and segment together, see kt.

8.2.3 Apparatus B: Force detection

The second apparatus, to demonstrate SS2, SS3, and SS4, would entail connecting two boards via CAT 5e cables. As seen in Figure 33: Apparatus B Configuration, one board (B) would send force measurements from SS2 (A) over the SS4 (C) and the other (D) would control an actuator (S) based on the received values.

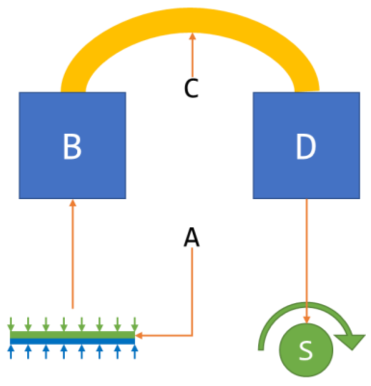


Figure 33: Apparatus B Configuration

A single short aluminium beam (x mm) kt was used as a platform for the test. Both boards were mounted to the beam. A CAT 5e cable was used to connect the two boards together.

8.2.4 PWM Control

To control the servomotors PWM control was required. A single PWM output channel was implemented in C for the STM32 Nucleo boards[111]. Much of the peripheral initialisation was completed used STM32CubeMX. The configuration file used can be found in the attached documents or in the file structure[112]. The configuration of the PWM, including GPIO used, can be found in Table 13: PWM Configuration.

Table 13: PWM Configuration

Parameter	Value
STM32F303k8 Pin	Port B Pin 4
Nucleo Connector Pin	D12
Timer	3
Channel	1
Prescaler (PSC - 16 bits value)	72
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits value)	100
Internal Clock Division (CKD)	No Division
Auto-reload preload	Disabled
PWM Generation Channel 1 Mode	PWM Mode 1

[111] For details relating to the microcontroller selection see 3.2.3 - Controller.
[112] See "\Code (C)\STMcubeMX - Apparatus A - Position.pdf", "\Code (C)\Apparatus A - Position_Kamikaze_Phoenix_Kamikaze_Phoenix.ioc", "\Code (C)\STMcubeMX - Apparatus B - Force.pdf", or "\Code (C)\Apparatus B - Force_Lucky_Boar_Lucky_Boar.ioc"

Once the PWM was initialised, the following methods was required for control:

1. Start the PWM[113];
2. Determine Duty Cycle;
3. Updated desired pulse width[114]; and,
4. Set the new pulse width & duty cycle[115].
5. Return to 2.

This could be used to control behaviour of the motors given their configuration[116], see Table 14: Motor PWM Requirements.

Table 14: Motor PWM Requirements

Parameter	Apparatus A (MG995) Value	Apparatus B (900-00008)Value
Cycle Period	20 ms (50 Hz)	20 ms (50 Hz)
Min Pulse (-60 Deg)	0.7 ms	1.3 ms
Max Pulse (+60 Deg)	2.5 ms	1.7 ms
Idle Pulse (0 Deg)	1.5 ms	1.5 ms

[113] See "\Code (C)\...\Drivers\STM32F3xx_HAL_Driver" - HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_1)

[114] See attached code "\Code (C)" or 18.7 - Set Duty Cycle

[115] See "\Code (C)\...\Drivers\STM32F3xx_HAL_Driver" - (__HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_1, pulse_width);

[116] The MG995 Datasheet neglects to inform the user of the duty cycle values accepted by the system. The values seen in Table 14 were empirically determined as the limits of the servomotor control.

9. Conclusion (10 pg)

9.1 Results and Discussion

The integrated system functioned as designed. Sensor readings from the force and proximity sensors were read by each controller then distributed amongst the system. The messaging and communication system worked as designed, and if a board lost power another could compensate.

Some slight improvements could be made to the general design:

- The mount structure should be redesigned to allow for easier access and removal. The mounts should also be reprinted[117], or the design should be revised;
- The CAT 5e and microUSB to USB cables worked as designed but were purchased in bulk according to the maximum expected distance between controllers and a power supply. These cables should be revised when dimensions are confirmed to remove unneeded slack; and,
- The power supply system could be improved upon slightly by using:
    - A single battery pack that could power all the controllers;
    - A single plug-in to power all the controllers; or,
    - Redesigning the power system so power is provided directly to the system (not via the NUCLEO boards).

The MCU selected was adequate however issues relating to firmware[118] may warrant changing to a PIC or Atmel chipset. The current system works, and further works would likely involve extending the capabilities of the existing firmware, so change may not actually be justified[119].

The integration of the system satisfied its requirements and the subsystems designed (SS1-SS5) were able to interface without problem.

[117] A cavity where the NUCLEO was meant to fit was 1mm (ish) too small as a result of 3D printing errors

[118] An example of the many many problems with STM's ecosystem:
The RX and TX communications worked on a NUCLEO board. This board could be connected to an FTDI chip, a logic analyser, an Arduino, a Bluetooth serial chip and work correctly. When this board was connected to an identical board it would crash. Weeks were wasted on this problem. Tried swapping to using an interrupt based UART system. Still crashed. Through it may have been a problem relating to pull up/pull down resistors. Nope. Thought it may have been a firmware problem. Nope. Or common ground. Nope. Swapped to a DMA based UART. Worked fine.
Never figured out what cause the problem. Couldn't find solutions online. Couldn't find references to the problem online. The problem was never solved.

[119] The underlying hardware interfacing has been completed, further works (to control actuators via the controllers, have a full body suit, etc..) would require changes to only the software portion of the code.

9.1.1 Lower Extremity Exoskeleton

The results of the lower extremity exoskeleton were inconclusive, as discussed in 1.3.1.2.1 - Variations, the mechanical structure and actuation portion of the project was not completed.

The perception systems designed were capable of measuring the position of the pilot and the force applied by/to the exoskeleton. Given a lower extremity exoskeleton to regulate, these sensors should allow for adequate feedback to control the system.

A controls structure via the transfer function method was created for a general 3 DOF RRR serial manipulator. Provided with actual values, this controls structure should provide PID values sufficiently close to optimal[120] to allow for proper control of a lower extremity exoskeleton.

Communication systems created allowed for the perception systems and control systems across an entire exoskeleton to coordinate. Properly mounted and connected, the systems designed should be able to correctly prescribe the desired actions for the actuator system.

Without testing the efficacy of the system cannot be commented on directly, however, all the major subsystems and the integrated system performed as designed and could be expected to perform as required if attached to a lower extremity exoskeleton.

9.1.2 Apparatus A

Apparatus A demonstrated the functionality of a 1 DOF exoskeleton. It demonstrated the functionality of the position sensing system but lacked the force sensing system.

The actuation system lacked the sufficient control to attain responsive or smooth control. Due to the position sensor range the suit had an insufficient distance and resolution to achieve the responsive control desired. The combination of these two factors resulted in the system being unstable with faster response times. Consequently, the P and D values in the PID control had to be reduced. The resulting system was slower than desired, closer to a crawl than a walk.

Increasing the quality of the perception systems would allow for a faster response time and more accurate control. However, to create a system capable of higher speeds would require an overall or replacement of the actuation system[121].

[120] Empirical tuning will refine initial values. The accuracy and precision of the initial values will dictate the difficulty of the tuning process.
[121] See 8 - Approach and execution: SS5

Despite its faults the system demonstrated a position-based control system controlling an exoskeleton in the desired manner. This was the great achievement of the project. While the full range of motions required for a full proof of concept where not demonstrated, the 1 DOF system displayed the fundamental concept of the exoskeleton functioning. Proximity was used to control a 1 DOF exoskeleton.

Further works will codify this, but ultimately Apparatus A proves the concept; proximity and position may be used to control a powered exoskeleton.

9.1.3 Apparatus B

Apparatus B performed as designed and demonstrated the force sensing system controlling the behaviour of an actuator. In a system were the torque output of actuators can be controlled, the system developed could be used to ensure correct and safe force output.

In the context of the hypothesis, Apparatus B demonstrated the functionality lacking in Apparatus A. Apparatus A demonstrated the position system controlling the actuators, Apparatus B demonstrated the communication system, the force sensors, and the control system cooperating. With the correct exoskeleton integrating and adapting the two systems would be trivial. Considering the results of Apparatus A and B holistically the desired functionality of the lower exoskeleton is achievable, the proximity based control system is feasible.

9.1.4 SS1

Under operating conditions, the effective range and resolution of the IR sensors was inadequate. As noted in 9.1.2 - Apparatus A, the system developed was capable of detecting the position of the user to the extent that the 1 DOF exoskeleton was able to maintain a constant offset. However, the sensor range was too limited for control. Dead bands existed when smaller pilot sections were use (e.g. the hand). This was likely a consequence of the intensity of the reflected IR being below the threshold of the phototransistor.

The original 3D printed mounts for the IR sensor PCBs were misshapen and did not fit as designed in the sensor frame. The original sensor frame was comprised entirely of aluminium, however, it proved too heavy for the system to lift, so a wood/aluminium solution was adopted.

The connection method (ribbon cable) was sufficient, by no provisions were made for how cables were to be managed on the mount (although provisions were made for cables on the exoskeleton frame, see attached documents[122]).

Natural IR from the sun and other sources added noise to the signal. This reduced the effective range of the sensors by creating a high-water mark where signals less than the ambient IR levels were lost[123]. Works have been done to remedy this, as noted in kt, but were not integrated into the final build.

9.1.5 SS2

The subsystem created was capable of generating a reading the force applied to both sets of sensors (internal and externals). These readings were accurate and could be used to control the behaviour of the systems actuators. Forces applied internally and externally were independent, and the system could be used to regulate force application.

When tested by Apparatus B, possible due to the force sensors being configured for a much lower range than rated[124], noise was present in the load cell readings. Further work should be done improve the signal quality from the force-based system. Testing conducted with an increased sampling time (10 seconds) yielded consistent and accurate readings, implying that the noise is constant kt

Within the design goals of measuring the force applied by the pilot to the exoskeleton and the exoskeleton to the environment the system was a success. Within the greater design goals of regulating force and ensuring safe and powerful operation of the system was a success.

9.1.6 SS3

Controls structures were created for three systems. The controls for the two 1 DOF systems were created via the second Ziegler-Nichols method, and the continuous rotation servomotor values were superseded by those found for the positional servomotor. As discussed in kt, the controls developed were effective at regulating the system.

However, the PID developed had a small integral gain and zero derivative gain. This implies that for stability the system had to be made insensitive to steady state error and sudden changes.

[122] See "\CAD Schematics\SMW-43219667-METR4901 - CABLE_MOUNT.pdf"
[123] Effectively creating a minimum signal strength threshold, and maximum detectable distance.
[124] Approximately 0-2 kg vs 50kg rating.

The low integral gain implies that the system had to be made insensitive to steady state error within the system. Upon further examination, when one side of the sensor array was in direct sunlight a constant error was present. With a large integral gain[125] the system would effectively suffer integral wind up from attempting to avoid the sun, and the entire system would be thrown off.

The zero derivative term implies that the system had to be completely desensitised to the fluctuations in signal intensity. This is likely the result of large amounts of meaningless fluctuations in signal intensity: noise. This is supported by the proportional gain of the system remaining large; were the changes in signal intensity large and sustained the proportional gain would also be affected.

The problems associated with the control system, therefore are not the result of the PID loop being inadequately tuned. Instead the PID has been tuned to a noisy signal and is behaving as best could be expected. Kt outlines refinements that could be made to the perception systems, which would result in a cleaner signal and smoother controls.

The third controls structure was created via the transfer function method. The general solution for a 3 DOF RRR serial manipulator was found which approximates the behaviour of each leg of the exoskeleton. The model created was compared to the results of known equations of motions, e.g. textbook solutions to other manipulators. When compared to a 2 DOF RR and 2 DOF PR manipulator (the code could be altered to suit any 1,2, or 3 DOF system) the result matched. So, to the extent that the transfer function was found for the system the 3 DOF solution is likely correct.

The most effective method of verifying the theory created would be to empirically test the values derived. However, as no actually values were confirmed for the exoskeleton the specific control parameters (PID values) could not be found.

In the context of developing control for the lower extremity exoskeleton, SS3 was not a success. Without confirmed values a control system could not be created beyond generalisation.

In the context of developing control for the 1 DOF exoskeleton, SS3 was a success. As discussed in kt, the system performed as instructed and operated correct. The feedback provided, and the control authority of the system limited the capability of the system. Were the

[125] Reasonable under normal conditions.

perception systems and actuation systems to be revised, it seems likely that the control methodology discussed in kt would result in stable and useful control of the exoskeleton.

9.1.7 SS4

SS4 was tested by daisy chaining multiple controller boards together via 1.5m CAT 5e cables. Each board would transmit the messages relevant to it on both UART 1 and UART 2. Messages received would be parse. Invalid messages were discarded. Valid messages where then passed along the chain to ensure all controller boards were informed, e.g. valid messages incoming to UART 1 were sent outgoing to UART 2.

For the thesis demonstration, two boards where connected and sensor readings from one board were used to communicate actuator commands to the other.

In both these test case the communication system was perfectly functioning. Messages were sent and transfer, no corrupt messages were interpreted as valid, and messages where interpreted correctly.

9.1.8 SS5

Analysis via logic analyser indicated that the PWM generation worked correctly in all circumstances. The PWM implemented worked correctly in all circumstances it was tested. The code developed was effective and functional.

When small angle changes are request the actuators responded unpredictably. This is due to the internal controls in the servomotor and the minimum resolution of the servomotor. On the edge of a defined edge the system would jittered and within the dead band of one unit of actuation not movement would occur. At higher speeds delays and overshoots occurred. With not feedback relating to the position, speed, or acceleration of the motor it was impossible to compensate for any of these errors. Additionally, safe force application could not be achieved as torque output could in no way be controlled.

Soldering loose cables to the through holes of a PCB header to control a system’s actuators is… not best practice. While later iterations of the PCB included dedicate servo headers, the PCBs used for demonstration featured fly-wires.

Areas of possible improvement considered, SS5 as a whole worked, and performed as desired.

The motors when tested in isolation or in Test A and Test B performed mostly as designed. While the response time of the MG995 was slower than preferred, it was adequate to demonstrate the functionality of perception systems.

9.2 Recommendations & Further Work

Recommendations and suggestions of further works to be completed for the system are as follows:

9.2.1 SS1

Ideally a permanent fastening method would be used for fasting the controls/perception system to the exoskeleton. A more robust method of attachment (while non-permeant) for consideration may be pipe clamps.

In further works, the mapping of physical coordinates and the controls may be updated to compensate for the true shape of human limbs.

Further work involving the proximity sensor should focus on increasing the accuracy, range, and precision of the system. Additional work should be done to increase the DOF measured by the system. Designs should be commissioned for low profile sensors that can be embedded in the contact areas or in tight areas (between fingers, legs, armpits, etc…).

The existing IR based design may be improved through the use of ultrasonic sensors, however they both use reflection-based technology that detects the profile of the pilot. Neither technology detects a fixed point on the user. In the case where a pilot rotates their joints, e.g. flipping their palm with an outstretched arm, the system is unable to detect changes in rotation. If the system models the 3D profile of the user it may be possible to determine the orientation of the pilot, however this inelegant, and requires extensive calibration for each pilot (and any changes in attire).

An improvement to the existing proximity sensing system may be to implement as system that actively emits a signal relating to the orientation of the pilot. Rather than a transceiver unit, the received remains on the suit and a transmitted it mounted on the pilot. The exoskeleton then seeks to follow and track the emitter. This solution would allow for rotational movements to be tracked and higher DOF systems to be controlled with less sensors.

As note in 13.5 - Magnetic Sensors, magnetic sensing methods were not used as they were not entirely in keeping with the “bubble” theme of the original designed. Two possible methods of emitter system are: Magnetic and IR Emitter systems[126,127].

[126] Internal suits would be constructed that feature magnetic (permanent or electromagnet) or IR (via photodiodes) emitters. Think TRON. It is possible to imaging a solution where magnets implanted in the pilots body are used to control an exoskeleton.

[127] Inductive sensors may also be a viable solution.

9.2.1.1 IR Sensor Refinements

The final system was commissioned and assembled, as shown in kt. When tested the position of the pilot could be found. As demonstrated during the thesis progress seminar distances within 10cm could be mapped accurately.

When tested in multiple environments it was discovered that while the TRCT5000 featured a "Daylight blocking filter" (Vishay Semiconductors, 2017) it was still affected by direct sunlight. This may have been due to the IR which was emitted by the sun. A possible solution to the ubiquity of IR in the testing environment may be to use a carrier wave to modulate the emitter frequency, and then use a bandpass filter to only receive the signals matching the carrier frequency.

To create the carrier, wave a 70kHz[128] PWM with a duty cycle of 50% was created, see 8.2.4 - PWM Control for details on how a 50kHz PWM was implemented for the servomotor interface. This was passed through an AND gate connected to the 5V rail (the PWM would signal the modulated power signal to the IR sensors, but would not power them). The AND gate selected was the SN74AHC1G08[129].

A sallen-key band pass filter was implemented to filter the 70kHz signal received by the IR sensor. The circuit used is detailed in kt and kt. The LM358AD was used as the Op-Amp for the filter, configured as seen in kt.

To increase the range of the sensors two methods may be considered. Firstly, using the internal gain of the sallen-key filter to shift the saturation range of the IR sensor, with the gain found in kt. Secondly, alternate sensors were considered. The OPB732[130] was selected as an alternate sensor option for the system. The OPB732 featured increased range characteristics and could be mounted to the existing IR sensor mount PCBs.

Components were ordered, and PCBs were fabricated, but did not arrive in time for construction. Further work should be done to test and improve these refinements.

[128] Preliminary analysis has shown that standard IR remote control systems use IR signals modulated between 30kHz and 56kHz (60kHz was observed). 70kHz should be free from interference, although additional analysis will be required. See attached documents as "TSOP17 - Photo Modules for PCM Remote Control Systems.pdf".

[129] The SN74AHC1G08 datasheet may be found in the attached documents as "SN74AHC1G08 - Single 2-Input Positive-AND Gate.pdf".

[130] The OPB732datasheet may be found in the attached documents as "OPB732 - Long Distance Reflective Switch.pdf".

9.2.2 SS2

Further work could be done to improve the accuracy of the sensors, possibly by replacing the YZC-161B with higher quality and consistency load cells. The rigid frame could be constructed from lighter materials[131] to reduce the burden on the actuation system; although for a full-strength exoskeleton aluminium may prove adequate. The load cell mounts should be reprinted, or recreated from a lighter material, matching the form factor of the sensors actually delivered. The rigid frame may also be altered to allow actuation within the foot[132].

For implementation in a lower extremity exoskeleton the load cells should be recalibrated for the expected range of force. To improve upon the system, it may be prudent to consider a fine and gross motor system. Given that contact with humans and everyday objects doesn’t exceed approximately 10kg of weight-force, areas contacting said items should be calibrated for a 0-10kg range. However, for an exoskeleton capable deadlifting a 500kg mass, sensors should also be calibrated for a range including 500kg. While it may be possible to simply use a load cell that encompasses the full range of force this may not present the resolution required. Instead, a solution to consider is separate gross and fine motor force sensors. One set calibrated for fine motor skills and delicate operations, one set calibrated for gross motor skills and heavy lifting.

9.2.3 SS3

The values derived for the system assume that all joints are revolute, disturbances and noise are minimal, and 1 DOF. The knee joint is not a revolute joint (a hinge) it is a modified hinge consisting of two joints (Chhajer, 2006). To accurately mimic the behaviour of the knee the exoskeleton employed may that also features a modified hinge joint. Consequentially the equations of motion developed would need to be revised.

The controls presume that each leg may be treated as independent, however, in the case of some extreme movements this may not be true. If the pilot stands on one leg and moves slowly, changes in position from the pilot adjusting their weight may be removed natural as steady state error. However, if the pilot stands on one leg and swings their other leg back and forth rapidly the supporting leg may be required to quickly compensate for error. The existing controls system may be sufficient, empirical testing is required to confirm. If it is insufficient, the controls devised may need to be revises to accommodate a 6 DOF serial manipulator.

[131] Suggestions include: moulded plastic, fibre glass, and carbon fibre.
[132] E.g. bending at the ball of the foot.

The controls system devised treats the velocity of the pilot as negligible. For the representative movements (standing, sitting, walking, squatting) this assumption is valid. As noted in 6.2.1 - Jacobian – Footnote 70, the integration of the pilots velocity is a simple addition to the existing controls of the system and may be relevant should the exoskeleton become capable of faster movement[133].

9.2.3.1 Emergency Stop

The control structure devised adequately performs in standard conditions; however, the stop condition for when the suit encounters an obstacle requires special treatment. Firstly, the existing controls architecture should be modified to detect high speed stop conditions[134]. Then, a secondary control system should be engineered for the immediate system response and minimum overshoot desired[135]. Control systems which in the presence of an external force actuate in the opposite direction should also be considered[136]. Finally, provisions should be made to assist the actuators in rapid deceleration, e.g. Back EMF braking, physical braking mechanisms, etc. To ensure the safety of operators and developers adequate safety features and emergency stopping methods are an absolute priority moving forwards.

9.2.4 SS4

The interface between the actuation system and the controls system was never completed. Provisions were made for connection to another system (these were also used for debugging) but were not tested as the actuation system was never completed. While one can be confident that the communication system would function as required, it cannot be said for certain without testing.

The provisions made in 8.2.4 - PWM Control allow for a single PWM signal to be output via controller boards. While additional inputs may be needed for feedback, it is possible for the exoskeleton actuation system to controlled via the existing controller boards without modification. As seen in Figure 11: Exoskeleton Design, there is a controller board per link, and therefore actuator. For further works it is recommended that works on the motor interface and the communications interface for the actuation system cease, as they have already been completed. Instead the existing controllers should be modified for the required feedback inputs

[133] E.g Running, jumping, etc…
[134] Instances were force applied to the external contact points increases rapidly.
[135] Continuing with a PID architecture, a large P, a small I, and a very large D term should result in a fast-rapid response.
[136] Current iterations of the force regulation systems restrict the torque output of the system, but do not actuate towards the user.

and the actuation side of the project should focus on fabricating a robust mechanical build with adequate feedback[137].

9.2.4.1 Communication speed

It is difficult to say if the communication system would have been fast enough to allow for real time control of the system. While messages were kept short and DMA was employed to speed messaging the baud rate of 9600 may have been simply too low for the communication speeds required.

Given an 8-bit char (ASCII character) approximately 109 messages (of standard size) could be sent per second, or about 9ms per message. Given the potential for a message to be passed on up to 6 times before being interpreted by the actuation system, and that up to 5 messages may be queued to be sent by each controller, a maximum delay of 192ms (21 message periods) could be expected. Given a human reaction time of 150 – 300ms (Yuhas, 2012), a system with a worst case reaction time of 192ms will lag behind the pilot and not react sufficiently quickly.

Going forward it is recommended that the baud rate of the system is raised to 115200 bits/s. Transmitting messages twelve times faster will result in a reaction time of 16ms and reduce the possibility of lag.

9.2.5 SS5

The attachment mechanism for mounting associated with the servomotors depended on cable ties. While cable ties are perfectly adequate for cable management, and function well as a stand in for hose clamps or pipe clamps at small diameters, they are not a robust or rigid method of fasting motors in place. In further iterations of the system it is strongly recommended that dedicated mounts for the servomotors be commissioned.

Ideally a permanent fastening method would be used for fasting the controls/perception system to the exoskeleton.

The exoskeleton segments, comprised of aluminium, were heavy and flexible. In Test A, a finger’s touch could cause the lower exoskeleton segment to wobble and flex. The system when tested would oscillate if accelerated too quickly, confusing the perception systems. This effectively capped the accuracy of the perception system and the speeds (and response times) at which stability could be attained.

[137] Position and torque are required, velocity is preferable.

Motor selection should not be determined by lead time. Instead actuators should be selected based their requirements. Originally the 900-00008 was to be used for Test A, but upon preliminary testing it was found that it lacked the torque to lift the lower exoskeleton segment beyond 30 degrees.

Neither servomotor allowed for control via torque or acceleration, the control variable used in SS3. Instead position and velocity were controlled. However, they could not be measured so the acceleration could not be indirectly controlled. The servomotors had their own internal control mechanisms, which had to be compensated for. Finally, the resolution of the MG995 was so low as to be visually perceptible. For all further iteration of the system it is strongly suggested that the actuators be reengineered.

The battery system used for SS5 was functional but ramshackle. An 8-battery receptacle was modified to fit 4 batteries to power the 6V MG995. This should be replaced with a 4-battery receptacle.

Preferential to the improvements discussed, SS5 should be make redundant. ***A functioning actuation and structural system would eliminate the need for SS5 entirely***.

9.2.6 General

As noted above[138], the quality of the actuation system and the perception systems is the limiting factor in the performance of the exoskeleton. Future work should focus on improving the precision and accuracy of the position sensors[139] and improving the actuation system.

Depending on the requirements and configuration of a larger exoskeleton, the daisy chain architecture may not scale. In further projects it may be prudent to consider upgrading some, if not all of the controller boards to a NUCLEO that supports 3 UART channels[140].

Within the scope of the project every major function requirement for the system was met. The demonstration rigs created, while outside of the original scope, demonstrate the functionality of the system and the feasibility of the project hypothesis. While further works would be required to create an exoskeleton with practical application, the works completed constitute a proof of concept for a novel proximity-based exoskeleton control system.

[138] See 9.1.4 - SS1 and 9.1.8 - SS5.
[139] Although improvements made (carrier wave, filter, new sensor, and amplifier) may prove adequate.
[140] A single master controller polling the other boards may also provide a viable solution.

9.3 Achievements

10.Bibliography

Agarwal, A. (2005). *Foundations of analog & digital electronic circuits* (1 ed.). Massachusetts: Massachusetts Institute of Technology. Retrieved June 1, 2018, from https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwiBu7yQl7LbAhUEoZQKHfYlBzkQFggpMAA&url=http%3A%2F%2Fsiva.bgk.uni-obuda.hu%2Fjegyzetek%2FMechatronikai_alapismeretek%2FEnglish_Mechatr%2FElectr_Eng-1%2FLiterature%2FFoundations%2520o

American Technologies Network Corporation. (2018, May 30). *How Does Night Vision Work*. Retrieved from atncorp.com: https://www.atncorp.com/hownightvisionworks

Arrow. (2018). *Magnetoresistive Sensor*. Retrieved May 30, 2018, from arrow.com: https://www.arrow.com/en/categories/sensors/magnetoresistive-sensors

Axe, D. (2012, May 23). *Combat Exoskeleton Marches Toward Afghanistan Deployment*. Retrieved May 30, 2018, from Wired: https://www.wired.com/2012/05/combat-exoskeleton-afghanistan/

Bulgrin, M. (2017, May 11). *The History of the Hose Clamp.* Retrieved from normagroup.com: https://blog.normagroup.com/en/the-history-of-the-hose-clamp/

Bunnings. (2018, May 31). *Kinetic 21 - 44mm 304 Stainless Steel Hose Clamp.* Retrieved May 31, 2018, from Bunnings.com: https://www.bunnings.com.au/kinetic-21-44mm-304-stainless-steel-hose-clamp_p4920194

Charara, S. (2015, July 9). *This robotic exoskeleton helps paralysed patients to walk and it's getting smarter*. Retrieved August 23, 2017, from Wearable: https://www.wareable.com/wearable-tech/exoskeleton-paralysed-patients-ekso-bionics-gt-sarah-thomas

Chhajer, B. (2006). *Anatomy of Knee.* New Delhi: Fusion Books.

Computer Cable Store. (2018, May 31). *11 7/8 Inch Black Standard Nylon Cable Tie - 100 Pack.* Retrieved May 31, 2018, from computercablestore.com: https://www.computercablestore.com/11-78-inch-black-standard-nylon-cable-tie-100-pack

Cornwall, W. (2015, October 15). *Feature: Can we build an 'Iron Man' suit that gives soldiers a robotic boost?* Retrieved August 20, 2017, from sciencemag.org: http://www.sciencemag.org/news/2015/10/feature-can-we-build-iron-man-suit-gives-soldiers-robotic-boost

Cracknell, A. P., & Hayes, L. (2007). *Introduction to Remote Sensing* (2 ed.). London: Taylor and Francis. Retrieved May 30, 2018

Cutnell, J. D., & Johnson, K. W. (1998). *Physics* (4th ed.). New York: Wiley.

Cyberdyne. (2015, August 1). *CYBERDYNE Inc. has begun seeking approval from the U. S. Food and Drug Administration (FDA)*. Retrieved August 23, 2017, from cyberdyne.jp: https://www.cyberdyne.jp/english/company/PressReleases_detail.html?id=1075

Cyberdyne. (2016). *What's HAL?* Retrieved August 19, 2017, from cyberdyne.jp: https://www.cyberdyne.jp/english/products/HAL/

Cybernetic Zoo. (2010, October 14). *1890 – Assisted-walking Device – Nicholas Yagn (Russian).* Retrieved May 30, 2018, from Cyberneticzoo.com: http://cyberneticzoo.com/tag/nicholas-yagn/

Cybernetic Zoo. (2010, April 10). *1965-71 – G.E. Hardiman I Exoskeleton – Ralph Mosher (American).* Retrieved May 30, 2018, from cyberneticzoo.com: http://cyberneticzoo.com/man-amplifiers/1966-69-g-e-hardiman-i-ralph-mosher-american/

Dawkins, P. (2018). *Differential Equations - Notes - Laplace's Equation*. Retrieved June 3, 2018, from Paul's Online Math Notes: http://tutorial.math.lamar.edu/Classes/DE/LaplacesEqn.aspx

Dorf, R. C., & Bishop, R. H. (2011). *Modern Control Systems* (12 ed.). Upper Saddle River: Pearson.

Dunietz, J. (2017, July 27). *Robotic Exoskeleton Adapts While It's Worn*. Retrieved August 20, 2017, from scientificamerican.com: https://www.scientificamerican.com/article/robotic-exoskeleton-ldquo-evolves-rdquo-while-its-worn/

Future Electronics. (2018, May 30). *What is Optoelectronics?* Retrieved from Future Electronics: http://www.futureelectronics.com/en/optoelectronics/infrared-receivers.aspx

Garbett, I. (2001, Janurary 1). *Light attenuation and exponential laws*. Retrieved May 30, 2018, from plus.maths.org: https://plus.maths.org/content/light-attenuation-and-exponential-laws

Golnaraghi, F., & Kuo, B. C. (2010). *Automatic Control Systems* (9 ed.). Hoboken: John Wiley & Sons, Inc. Retrieved June 4, 2018

Gross, K. (2018, Feburary 19). *Ultrasonic Sensors: Advantages and Limitations*. Retrieved May 30, 2018, from MaxBotix: https://www.maxbotix.com/articles/advantages-limitations-ultrasonic-sensors.htm/

Horowitz, P., & Hill, W. (2015). *The Art of Electronics* (3 ed.). Cambridge: Cambridge University Press.

Jackson, R., Green, K. R., & Eisenbeis, R. (2017). *Achieve greater precision, reliability with integrated magnetic sensing technology*. Retrieved May 30, 2018, from ti.com: http://www.ti.com/general/docs/lit/getliterature.tsp?baseLiteratureNumber=sszy030&fileType=pdf

Karlin, S. (2011, July 29). *Raytheon Sarcos's Exoskeleton Nears Production*. Retrieved August 11, 2017, from spectrum.ieee.org: http://spectrum.ieee.org/at-work/innovation/raytheon-sarcoss-exoskeleton-nears-produc

Keller, M. (2016, August 25). *Exoskeleton - Do You Even Lift, Bro? Hardiman Was GE's Muscular Take On The Human-Machine Interface*. (General Electric) Retrieved May 30, 2018, from GE Reports: https://www.ge.com/reports/do-you-even-lift-bro-hardiman-and-the-human-machine-interface/

Keyence Corporation. (2018). *What is a Inductive Proximity Sensor?* Retrieved May 30, 2018, from keyence.com: https://www.keyence.com/ss/products/sensor/sensorbasics/proximity/info/

Khatib, O. (2008). Chapter 5 - Dynamics. In O. Khatib, *Introduction to Robotics* (pp. 125-150). Stanford: Stanford University.

Liew, S. C. (2018, May 30). *Electromagnetic Waves*. Retrieved from Centre for Remote Imaging, Sensing and Processing.: https://crisp.nus.edu.sg/~research/tutorial/em.htm

Lynch, D. K., & Livingston, W. C. (2001). *Color and Light in Nature* (2nd ed.). Cambridge, United Kingdom: Cambridge University Press. Retrieved May 30, 2018, from https://books.google.com.au/books?id=4Abp5FdhskAC&pg=PA231&redir_esc=y#v=onepage&q&f=false

Merriam-Webster Dictionary. (2018, May 18). *noise*. Retrieved May 30, 2018, from merriam-webster.com: https://www.merriam-webster.com/dictionary/noise

National Instruments. (2018). *PID Theory Explained.* Retrieved June 4, 2018, from NationalInstruments.com: http://www.ni.com/white-paper/3782/en/

Nave, C. R. (2017). *Operational Amplifiers*. (Georgia State University, Department of Physics and Astronomy) Retrieved June 4, 2018, from hyperphysics.phy-astr.gsu.edu: http://hyperphysics.phy-astr.gsu.edu/hbase/Electronic/opamp.html#c1

Ogata, K. (2010). *Modern Control Engineering* (2 ed.). New Jersey, United States of America: Prentice Hall. Retrieved August 25, 2017

Otaga, K. (2004). *System Dynamics* (4 ed.). Upper Saddle River: Pearson. Retrieved June 4, 2018

Robomart. (2015, November 9). *Advantages and Disadvantages of ultrasonic distance sensor*. Retrieved May 30, 2018, from Robomart: http://roboticsensors.blogspot.com/2015/11/advantages-and-disadvantages-of.html

Siciliano, B., & Khatib, O. (2016). *Springer Handbook of Robotics* (2 ed.). (B. Siciliano, & O. Khatib, Eds.) Berlin: Springer Nature. doi:10.1007/978-3-319-32552-1

Texas Instruments Incorporated. (2017). *Hall effect sensors*. Retrieved May 30, 2018, from ti.com: http://www.ti.com/sensing-products/magnetic-sensors/hall-effect/overview.html

Thomas Publishing Company. (2018). *Capacitive Proximity Sensors*. Retrieved May 30, 2018, from Thomas: https://www.thomasnet.com/articles/instruments-controls/proximity-sensors

TT Electronics/Optek Technology. (2018). *TT Electronics/Optek Technology OPB732*. Retrieved June 5, 2018, from digikey.com: https://www.digikey.com/product-detail/en/tt-electronics-optek-technology/OPB732/365-1691-ND/1637069

Vishay Semiconductors. (2017, February 8). TCRT5000 - Reflective Optical Sensor with Transistor Output.

Yagin, N. (1890, February 11). *United States of America Patent No. 440684.*

Yuhas, D. (2012, May 24). *Speedy Science: How Fast Can You React?* Retrieved from scientificamerican.com: https://www.scientificamerican.com/article/bring-science-home-reaction-time/

ZJIA. (2018, June 1). *Generic YZC-161B 50kg Body Scale Sensor Human Scale Weighing Load Cell Sensor (Pack of 4)* . Retrieved June 1, 2018, from Amazon.com: https://www.amazon.com/Generic-YZC-161B-Scale-Sensor-Weighing/dp/B00MTJ6WZ2

11. Appendix A: Representative Movements

11.1 Level One Functionality: Standing

To stand while the exoskeleton system is engaged requires the system to be capable of achieving equilibrium and control in a static environment.

Level one functionality demonstrates that for an instantaneous snapshot of operation that the system is capable of regulated operation. Note, level one functionality may also highlight the system's ability to compensate for steady state error.

11.2 Level Two Functionality: Squatting

Level two functionality requires level one functionality.

To squat while the exoskeleton system is engaged requires the system to be capable of control in a dynamic environment where the pilot is moving. A squat allows for the pilot to engage in motion at the stable pace of the exoskeleton, and as such may non-real-time operations.

Level two functionality demonstrates that the system is capable on a fundamental level of mirroring the pilot's movements.

11.3 Level Three Functionality: Stair Climbing

Level three functionality requires level two functionality.

To climb up stairs while the exoskeleton system is engaged requires the system to be capable of control in a dynamic environment where the pilot is moving while also applying force to the environment. However, should the system apply too much force to the environment the exoskeleton will simply lift itself off the ground, ultimately not requiring meaningful force regulation.

Level three functionality demonstrates that the system is capable of applying force to an environment.

11.4 Level Four Functionality: Sitting

Level four functionality requires level three functionality.

To sit down while the exoskeleton system is engaged requires the system to be capable of control in a dynamic environment where the pilot is moving while also applying force to the environment in a regulated manner. If the suit applied too great a force to a seat, then it may damage the seat. If the system is incapable of allowing the pilot to rest on the system, it may result in uncontrolled behaviour. As the pilot sits the system should concede to the force applied by the seat, until the point at which the plot applies force to the upper thighs of the system.

Simply, if a suit is capable of sitting, it is capable of interacting with the environment without destroying. Level four functionality demonstrates that the system is capable of applying force to an environment in a safe and regulated manner.

11.5 Level Five Functionality: Standing/Walking/Sprinting

Level five functionality requires level four functionality.

Presuming all prior levels of functionality are attained the suit should be capable of all required actions. However, to switch contexts and move between standing, moving, and running actions requires dynamic real time control. For an exoskeleton system to be truly viable, it is essential that context switching, and real time control are possible.

Level five functionality demonstrates that the system is capable of acting in a real environment and acts as a complete proof of concept for position-based control methods.

12.Appendix B: Prior Art

The following outlines some current developments in exoskeleton technologies.

12.1 HULC

The Human Universal Load Carrier (HULC) is battery-powered lower extremity exoskeleton initially developed by Berkeley Robotics and Human Engineering Laboratory, before entering an exclusive licensing agreement with Lockheed Martin in 2009 (Axe, 2012). The system uses hydraulics to amplify the pilot's knees and hips while supporting a load of up to 90kg. Designed for military applications it claims six hours of battery and uses force-based sensors for control.

The HULC was abandoned as" it proved impractical, exhausting pilots instead of supercharging them" (Cornwall, 2015) and has been succeeded by the TALOS project (Cornwall, 2015).

12.2 EskoGT

In 2010 the original developer of the HULC, Esko Bionics revealed the Exoskeleton Lower Extremity Gait System (eLEGS) (Charara, 2015). With a maximum battery life of 6 hours and maximum gait of 3.2m/s (Charara, 2015), the system uses pushbuttons and force-motion sensors for control. Specially design for medical applications, the exoskeleton uses preprogramed movements to aid the mobility of stroke and spinal injury patients.

The suit is ill suited for dynamic environments as its finite range of movements prohibits uneven surfaces. While the suit may assist those with "upper extremity motor function of at least 4/5 in at least one arm" (Charara, 2015), the suit is slower than a wheelchair and is not an improvement on standard human movement.

12.3 Raytheon XOS Exoskeleton

The 2008 Raytheon XOS Exoskeleton developed by Raytheon is a full body exoskeleton that can support up to 23kg on each arm (Karlin, 2011). The suit uses force-based sensors for control. Despite claims that the exoskeleton would be ready for production by 2016, they have made no public comments on progress since 2011 (Karlin, 2011).

12.4 Warrior Web

The Warrior Web non-rigid exoskeleton was first demonstrated at the 2016 DARPA Demo Day (Cornwall, 2015). Developed by DARPA, it used preprogramed commands to assist with the pilot's ankle motions. However, it was unpredictable in uneven terrain, malfunctioned, and could not transition readily between a walking and running state (Cornwall, 2015).

12.5 Hybrid Assistive Limb (HAL)

In 1997 Cyberdyne unveiled the Hybrid Assistive Limb (HAL) (Cyberdyne, 2016). The HAL's iterations include a battery-powered lower extremity exoskeleton and a full body exoskeleton (Cyberdyne, 2016). Through a combination of bioelectrical sensors and force sensors the HAL measured muscle contracts to trigger preprogramed movements.

The system has had mixed success, and despite applying for USA FDA approval in 2014, the HAL is yet to be permitted for use in the US (Cyberdyne, 2015).

13.Appendix C: Proximity Sensors

13.1 IR Transceiver

Infrared light, or IR, is a form of electromagnetic (EM) radiation general not visible to human eye's (Lynch & Livingston, 2001). The wavelength of IR is typically defined as ranging from 700 nanometres (430 THz) to 1 millimetre (300 GHz) (Liew, 2018). IR is emitted by the sun, artificial lighting, fires, as thermal radiation from objects (and animals), and from IR emitters (American Technologies Network Corporation, 2018).

The prototypical IR emitter is a light emitting diode (LED) composed to emit IR when power. They typically share a formfactor with standard LEDs and are often used in IR communication. To receive a signal transmitted via an IR emitter and IR received is used. IR receivers may take the form of a photoresistor configured for IR range light. IR emitters and receivers are often used in concert to transmits a message (via the emitter) and then receive it (via the receiver) (Future Electronics, 2018).

Like all EM waves, IR suffers from attenuation (Garbett, 2001) and is capable of being reflected off a non-absorbing material. As seen in Figure 34, by emitting IR (A) and measuring the intensity of the light reflected (B) it is possible to determine the distance from the reflective surface (C) and the emitter. This principle may be applied to determine the distance of an object from a transceiver (IR emitter/transmitter and receiver).

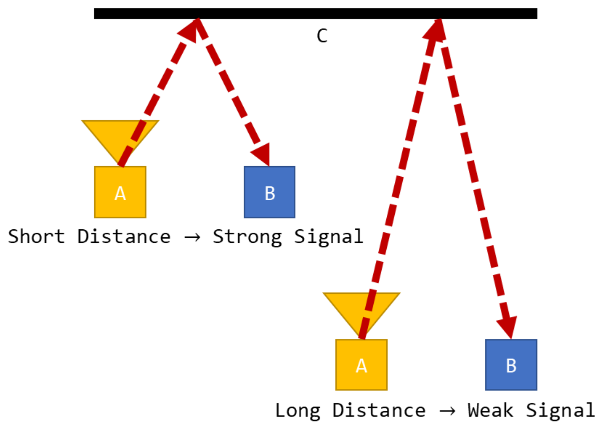


Figure 34: IR Proximity Sensing

IR is an effective method of detecting range, in fact IR is often employed in LiDAR (Cracknell & Hayes, 2007). Assuming line of sight exists between the reflection point and the IR receiver there is no minimum range. Additionally, IR technology is small, affordable, and ubiquitous.

Under ideal conditions an IR transceiver would be capable of perceiving the instance between an exoskeleton and its pilot.

Outside of ideal conditions complications with IR technology can occur. As noted above, IR is ubiquitous and is emitted by the sun, artificial lighting, and animals meaning that even if all undesirable frequencies where filtered from an IR signal noise may still exist. Under poor operating conditions an IR transceiver may be saturated with IR rendering it effectively blind. Additionally, variability in the reflective surface may result in IR being reflected inconsistently or not at all. Under these conditions mapping from signal intensity to distance may be impossible, as different surfaces will yield different signal intensities.

13.2 Ultrasonic Range Finders

Ultrasonic waves are sound waves with a frequency above the audible range of humans, approximately 20 kHz (Cutnell & Johnson, 1998). Ultrasonic waves can used for range finding by emitting an ultrasonic sound and recording the time for the wave to be reflected back. Ultrasonic range finders have been used as the autofocus in cameras, and motion detectors, and are the underlying technology for Sonar.

Ultrasonic ranger sensors have the advantages of (Gross, 2018):

- not being dependant on the lighting conditions and offering reasonably high resolution at short distances; and,
- using sound rather than light, ultrasonic range finders are adept at detecting clear or transparent objects.

However, ultrasonic range finders have limitations (Robomart, 2015):

- they feature a minimum effective range, preventing their noncontact use at close range;
- the transmission of ultrasonic waves is affected by temperatures, humidity, and airborne particles; altering the perceived distance;
- for accurate measurement they require a hard, flat, level surface directly opposite and perpendicular. Compared to the irregular shapes and the hair of the human form, they may be ill suited; and,
- they are effect by ambient acoustic noise. The operation of the exoskeleton itself (specifically actuators) may create sufficient noise to interfere with any ultrasonic range finding.

13.3 Capacitive Proximity Sensors

Capacitive proximity sensors act in the manner of a capacitor where one plate functions as an output or a switch (Thomas Publishing Company, 2018). Capacitive proximity sensors are effective in high precision applications and controlled environments; however, they are less effective at greater ranges. Given the possibility of large uneven surfaces, comprised of unspecified materials capacitive proximity sensors were neglected from further consideration (Thomas Publishing Company, 2018).

13.4 Inductive Proximity Sensors

Inductive proximity sensors operate by the induction of eddy currents in metals and similarly conductive materials (Keyence Corporation, 2018). Humans are not metals or similarly conductive material, and as such not suitable for range finding via inductive proximity sensors (Keyence Corporation, 2018).

13.5 Magnetic Sensors

Range finding is possible using hall effect sensors (Texas Instruments Incorporated, 2017), magnetometers (Jackson, Green, & Eisenbeis, 2017), and Magnetoresistive Sensors (Arrow, 2018). While future iteration of the exoskeleton may include proximity detection based on magnetic sensors, they all depend on magnets (permanent or otherwise) to generate a field to be measured. In keeping the spirit of the "*Bubble*" design of the exoskeleton, it was elected to avoid perception methods that require sensors to be mounted to the pilot, and therefore, all magnetic sensors were excluded from selection.

14.Appendix D: Control Outputs

14.1 Equations of Motion

Table 15: Equations of Motion

T1	dda2*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + L1*cos(a2)*L2 + l3^2 + L1*cos(a2 + a3)*l3) + l2*m2*(l2 + L1*cos(a2))) + dda3*(Izz3 + l3*m3*(l3 + L1*cos(a2 + a3) + L2*cos(a3))) - da1*(L1*da2*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) - da2*(L1*(da1 + da2)*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) + dda1*(Izz1 + Izz2 + Izz3 + L1^2*m2 + L1^2*m3 + L2^2*m3 + l1^2*m1 + l2^2*m2 + l3^2*m3 + 2*L1*l3*m3*cos(a2 + a3) + 2*L1*L2*m3*cos(a2) + 2*L1*l2*m2*cos(a2) + 2*L2*l3*m3*cos(a3)) + g*m2*(l2*cos(a1 + a2) + L1*cos(a1)) + g*m3*(L2*cos(a1 + a2) + L1*cos(a1) + l3*cos(a1 + a2 + a3)) + g*l1*m1*cos(a1) - da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))*(da1 + da2 + da3)
T2	dda1*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + L1*cos(a2)*L2 + l3^2 + L1*cos(a2 + a3)*l3) + l2*m2*(l2 + L1*cos(a2))) + da1*(L1*da1*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) - L2*da3*l3*m3*sin(a3)) + dda2*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + l3^2) + l2^2*m2) + dda3*(Izz3 + l3*m3*(l3 + L2*cos(a3))) + g*m3*(L2*cos(a1 + a2) + l3*cos(a1 + a2 + a3)) + g*l2*m2*cos(a1 + a2) - L2*da2*da3*l3*m3*sin(a3) - L2*da3*l3*m3*sin(a3)*(da1 + da2 + da3)
T3	dda3*(m3*l3^2 + Izz3) + dda1*(Izz3 + l3*m3*(l3 + L1*cos(a2 + a3) + L2*cos(a3))) + da1*(da1*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3)) + L2*da2*l3*m3*sin(a3)) + dda2*(Izz3 + l3*m3*(l3 + L2*cos(a3))) + g*l3*m3*cos(a1 + a2 + a3) + L2*da2*l3*m3*sin(a3)*(da1 + da2)

Where

$$ai = \theta_i$$

$$dai = \dot{\theta}_i$$

$$ddai = \ddot{\theta}_i$$

$$Li = Length\ of\ Link$$

$$li = Distance\ from\ joint\ to\ center\ of\ mass\ of\ link$$

14.2 Laplace Transformed Equations of Motion

Table 16: Laplace EOM

T1	dda2*(m3*L2^2 + 2*m3*cos(a3)*L2*l3 + L1*m3*cos(a2)*L2 + m2*l2^2 + L1*m2*cos(a2)*l2 + m3*l3^2 + L1*m3*cos(a2 + a3)*l3 + Izz2 + Izz3) + dda3*(Izz3 + l3^2*m3 + L1*l3*m3*cos(a2 + a3) + L2*l3*m3*cos(a3)) - da2*(L1*(da2 + A1*s)*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) - A1*s*(L1*da2*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) + A1*s^2*(Izz1 + Izz2 + Izz3 + L1^2*m2 + L1^2*m3 + L2^2*m3 + l1^2*m1 + l2^2*m2 + l3^2*m3 + 2*L1*l3*m3*cos(a2 + a3) + 2*L1*L2*m3*cos(a2) + 2*L1*l2*m2*cos(a2) + 2*L2*l3*m3*cos(a3)) + (A1*g*m3*(L1*s - l3*sin(a2 + a3) - L2*sin(a2) + l3*s*cos(a2 + a3) + L2*s*cos(a2)))/(s^2 + 1) + (A1*g*m2*(L1*s - l2*sin(a2) + l2*s*cos(a2)))/(s^2 + 1) - da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))*(da2 + da3 + A1*s) + (A1*g*l1*m1*s)/(s^2 + 1)
T2	dda1*(Izz2 + Izz3 + m3*(L2^2 + l3^2 + 2*L2*l3*cos(a3) + (A2*L1*L2*s)/(s^2 + 1) - (A2*L1*l3*(sin(a3) - s*cos(a3)))/(s^2 + 1)) + l2*m2*(l2 + (A2*L1*s)/(s^2 + 1))) + da1*(L1*da1*((A2*l2*m2)/(s^2 + 1) + (A2*L2*m3)/(s^2 + 1) - (A2*l3*m3*(cos(a3) - s*sin(a3)))/(s^2 + 1)) - L2*da3*l3*m3*sin(a3)) + dda3*(Izz3 + l3*m3*(l3 + L2*cos(a3))) + A2*s^2*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + l3^2) + l2^2*m2) - g*m3*((A2*L2*(sin(a1) - s*cos(a1)))/(s^2 + 1) + (A2*l3*(sin(a1 + a3) - s*cos(a1 + a3)))/(s^2 + 1)) -

	(A2*g*l2*m2*(sin(a1) - s*cos(a1)))/(s^2 + 1) - L2*da3*l3*m3*sin(a3)*(da1 + da3 + A2*s) - A2*L2*da3*l3*m3*s*sin(a3)
T3	dda2*(Izz3 + l3*m3*(l3 + (A3*L2*s)/(s^2 + 1))) + da1*(da1*l3*m3*((A3*L2)/(s^2 + 1) - (A3*L1*(cos(a2) - s*sin(a2)))/(s^2 + 1)) + (A3*L2*da2*l3*m3)/(s^2 + 1)) + dda1*(Izz3 + l3*m3*(l3 - (A3*L1*(sin(a2) - s*cos(a2)))/(s^2 + 1) + (A3*L2*s)/(s^2 + 1))) + A3*s^2*(m3*l3^2 + Izz3) - (A3*g*l3*m3*(sin(a1 + a2) - s*cos(a1 + a2)))/(s^2 + 1) + (A3*L2*da2*l3*m3*(da1 + da2))/(s^2 + 1)

14.3 Transfer Functions

Table 17: Transfer Functions

$\frac{\theta_1}{\tau_1}$	-1/(((da2*(L1*da2*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) - dda2*(m3*L2^2 + 2*m3*cos(a3)*L2*l3 + L1*m3*cos(a2)*L2 + m2*l2^2 + L1*m2*cos(a2)*l2 + m3*l3^2 + L1*m3*cos(a2 + a3)*l3 + Izz2 + Izz3) - dda3*(Izz3 + l3^2*m3 + L1*l3*m3*cos(a2 + a3) + L2*l3*m3*cos(a3)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))*(da2 + da3))/(s^2*(Izz1 + Izz2 + Izz3 + L1^2*m2 + L1^2*m3 + L2^2*m3 + l1^2*m1 + l2^2*m2 + l3^2*m3 + 2*L1*l3*m3*cos(a2 + a3) + 2*L1*L2*m3*cos(a2) + 2*L1*l2*m2*cos(a2) + 2*L2*l3*m3*cos(a3)) - s*(L1*da2*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) - L1*da2*s*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + (g*m3*(L1*s - l3*sin(a2 + a3) - L2*sin(a2) + l3*s*cos(a2 + a3) + L2*s*cos(a2)))/(s^2 + 1) + (g*m2*(L1*s - l2*sin(a2) + l2*s*cos(a2)))/(s^2 + 1) - da3*l3*m3*s*(L1*sin(a2 + a3) + L2*sin(a3)) + (g*l1*m1*s)/(s^2 + 1)) - 1)*(s^2*(Izz1 + Izz2 + Izz3 + L1^2*m2 + L1^2*m3 + L2^2*m3 + l1^2*m1 + l2^2*m2 + l3^2*m3 + 2*L1*l3*m3*cos(a2 + a3) + 2*L1*L2*m3*cos(a2) +

	2*L1*l2*m2*cos(a2) + 2*L2*l3*m3*cos(a3)) - s*(L1*da2*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + da3*l3*m3*(L1*sin(a2 + a3) + L2*sin(a3))) - L1*da2*s*(l2*m2*sin(a2) + l3*m3*sin(a2 + a3) + L2*m3*sin(a2)) + (g*m3*(L1*s - l3*sin(a2 + a3) - L2*sin(a2) + l3*s*cos(a2 + a3) + L2*s*cos(a2)))/(s^2 + 1) + (g*m2*(L1*s - l2*sin(a2) + l2*s*cos(a2)))/(s^2 + 1) - da3*l3*m3*s*(L1*sin(a2 + a3) + L2*sin(a3)) + (g*l1*m1*s)/(s^2 + 1)))
$\frac{\theta_2}{\tau_2}$	1/(((dda1*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + l3^2) + l2^2*m2) + dda3*(Izz3 + l3*m3*(l3 + L2*cos(a3))) - L2*da1*da3*l3*m3*sin(a3) - L2*da3*l3*m3*sin(a3)*(da1 + da3))/(dda1*(m3*((L1*l3*(sin(a3) - s*cos(a3)))/(s^2 + 1) - (L1*L2*s)/(s^2 + 1)) - (L1*l2*m2*s)/(s^2 + 1)) - s^2*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + l3^2) + l2^2*m2) + g*m3*((L2*(sin(a1) - s*cos(a1)))/(s^2 + 1) + (l3*(sin(a1 + a3) - s*cos(a1 + a3)))/(s^2 + 1)) - L1*da1^2*((L2*m3)/(s^2 + 1) + (l2*m2)/(s^2 + 1) - (l3*m3*(cos(a3) - s*sin(a3)))/(s^2 + 1)) + (g*l2*m2*(sin(a1) - s*cos(a1)))/(s^2 + 1) + 2*L2*da3*l3*m3*s*sin(a3)) - 1)*(dda1*(m3*((L1*l3*(sin(a3) - s*cos(a3)))/(s^2 + 1) - (L1*L2*s)/(s^2 + 1)) - (L1*l2*m2*s)/(s^2 + 1)) - s^2*(Izz2 + Izz3 + m3*(L2^2 + 2*cos(a3)*L2*l3 + l3^2) + l2^2*m2) + g*m3*((L2*(sin(a1) - s*cos(a1)))/(s^2 + 1) + (l3*(sin(a1 + a3) - s*cos(a1 + a3)))/(s^2 + 1)) - L1*da1^2*((L2*m3)/(s^2 + 1) + (l2*m2)/(s^2 + 1) - (l3*m3*(cos(a3) - s*sin(a3)))/(s^2 + 1)) + (g*l2*m2*(sin(a1) - s*cos(a1)))/(s^2 + 1) + 2*L2*da3*l3*m3*s*sin(a3)))
$\frac{\theta_3}{\tau_3}$	1/(((dda1*(m3*l3^2 + Izz3) + dda2*(m3*l3^2 + Izz3))/(s^2*(m3*l3^2 + Izz3) + da1*(da1*l3*m3*(L2/(s^2 + 1) - (L1*(cos(a2) - s*sin(a2)))/(s^2 + 1)) + (L2*da2*l3*m3)/(s^2 + 1)) - dda1*l3*m3*((L1*(sin(a2) - s*cos(a2)))/(s^2 + 1) - (L2*s)/(s^2 + 1)) - (g*l3*m3*(sin(a1 + a2) - s*cos(a1 + a2)))/(s^2 + 1) + (L2*da2*l3*m3*(da1 + da2))/(s^2 + 1) +

	(L2*dda2*l3*m3*s)/(s^2 + 1)) + 1)*(s^2*(m3*l3^2 + Izz3) + da1*(da1*l3*m3*(L2/(s^2 + 1) - (L1*(cos(a2) - s*sin(a2)))/(s^2 + 1)) + (L2*da2*l3*m3)/(s^2 + 1)) - dda1*l3*m3*((L1*(sin(a2) - s*cos(a2)))/(s^2 + 1) - (L2*s)/(s^2 + 1)) - (g*l3*m3*(sin(a1 + a2) - s*cos(a1 + a2)))/(s^2 + 1) + (L2*da2*l3*m3*(da1 + da2))/(s^2 + 1) + (L2*dda2*l3*m3*s)/(s^2 + 1)))

15.Appendix E: Equations of Motion

15.1 Explicit Form of the Equations of Motion

We begin with the Euler–Lagrange equations, or Lagrange's equations of the second kind, Equation 17.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = \tau$$

Equation 17: Euler–Lagrange equations (Khatib, 2008)

Where τ is the vector of applied generalised torques. The Lagrangian, L, is given by Equation 18.

$$L = K - V$$

Equation 18: The Lagrangian (Khatib, 2008)

Where V is the potential energy of the system, and K is the kinetic energy of the system. As seen in Equation 19, K may be given in terms of the generalised velocities, $\dot{q}$ (as seen in Equation 17: Euler–Lagrange equations) and the manipulator mass matrix M.

$$K = \frac{1}{2}\dot{q}^T M \dot{q}$$

Equation 19: Kinetic Energy (Khatib, 2008)

Through Equation 17 we may say that $\frac{\partial K}{\partial \dot{q}} = \frac{\partial}{\partial \dot{q}}\left(\frac{1}{2}\dot{q}^T M(q)\dot{q}\right) = M\dot{q}$

Equation 20 and Equation 21 hold.

$$\frac{\partial K}{\partial \dot{q}} = \frac{\partial}{\partial \dot{q}}\left(\frac{1}{2}\dot{q}^T M(q)\dot{q}\right) = M\dot{q}$$

Equation 20: Kinetic Energy Partial Derivative

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}}\right) = M\ddot{q} + \dot{M}\dot{q}$$

Equation 21: Kinetic Energy Time Derivative

Thus the inertial forces of Equation 18 may be expressed as Equation 22, where v is the vector of centrifugal and Coriolis forces, Equation 23.

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}}\right) - \frac{\partial K}{\partial q} = M\ddot{q} + \dot{M}\dot{q} - \frac{1}{2}\begin{bmatrix} \dot{q}^T \dfrac{\partial M}{\partial q_1}\dot{q} \\ \vdots \\ \dot{q}^T \dfrac{\partial M}{\partial q_n}\dot{q} \end{bmatrix} = M\ddot{q} + v(q, \dot{q})$$

Equation 22: Inertial Forces

$$v(q, \dot{q}) = \dot{M}\dot{q} - \frac{1}{2}\begin{bmatrix} \dot{q}^T \dfrac{\partial M}{\partial q_1}\dot{q} \\ \vdots \\ \dot{q}^T \dfrac{\partial M}{\partial q_n}\dot{q} \end{bmatrix} = C(q)[\dot{q}^2] + B(q)[\dot{q}\dot{q}]$$

Equation 23: Vector of centrifugal and Coriolis forces

Through Equation 22 we may yield the explicit form of the equations of motion (EOM), see Equation 24. Where g is the vector of gravity force and v is the vector of centrifugal and Coriolis forces. Equation 24, once found, may be used to map the relationship between the torque applied by the systems actuators and the resulting system configuration.

$$M(q)\ddot{q} + v(q, \dot{q}) + g(q) = \tau$$

Equation 24: Explicit form of EOM

Equation 24 forms the basis of finding the transfer function for the system.

15.1.1 Jacobian

For a system, in this case a manipulator, in the configuration given by the vector q there is corresponding position for the end-effector given by the vector x. The Jacobian matrix, J, describes the relationship between the time derivatives of q and x ($\dot{q}$ and $\dot{x}$ respectively). The Jacobian matrix, or simply the Jacobian, given by Equation 25, allows use to describe the system by Equation 26, Where $q = (\theta_1, \theta_2, \theta_3)$..

$$J = \begin{pmatrix} \dfrac{\partial x}{\partial \theta_1} & \dfrac{\partial x}{\partial \theta_2} & \dfrac{\partial x}{\partial \theta_3} \\ \dfrac{\partial y}{\partial \theta_1} & \dfrac{\partial y}{\partial \theta_2} & \dfrac{\partial y}{\partial \theta_3} \\ \dfrac{\partial z}{\partial \theta_1} & \dfrac{\partial z}{\partial \theta_2} & \dfrac{\partial z}{\partial \theta_3} \end{pmatrix}$$

Equation 25: Jacobian

$$\dot{x} = J(q)\dot{q}$$

Equation 26: Relationship between q and x

15.1.2 Explicit form of Manipulator Mass Matrix

Kinetic energy is subject to the additive property (Siciliano & Khatib, 2016), and thus the total kinetic energy of a system is the summation of the kinetic energy of its links[141].

The kinetic energy of each link is comprised of a rotational and linear motion component. For a link with linear motion of v_{C_i}, an angular motion of ω_i, and an inertia tensor of I_{C_i}, the kinetic energy of the link , K_i, is given by Equation 27Equation 27. Where C_i refers to the centre of mass of the link.

$$K_i = \frac{1}{2}\left(m_i v_{C_i}^T v_{C_i} + \omega_i^T I_{C_i} \omega_i\right) = \frac{1}{2}\left(m_i \dot{q}^T J_{vi}^T J_{vi} \dot{q} + \dot{q}^T J_{\omega i}^T I_{C_i} J_{\omega i} \dot{q}\right)$$

Equation 27: Kinetic Energy of Link i

Given Equation 27 and the additive property it may be said that the kinetic of the system in total is given by Equation 28,

$$K = \sum_{i=1}^{n} K_i$$

Equation 28: Kinetic energy of System

$$K = \frac{1}{2}\sum_{i=1}^{n}\left(m_i \dot{q}^T J_{vi}^T J_{vi} \dot{q} + \dot{q}^T J_{\omega i}^T I_{C_i} J_{\omega i} \dot{q}\right) = \frac{1}{2}\dot{q}^T \left[\sum_{i=1}^{n}\left(m_i J_{vi}^T J_{vi} + J_{\omega i}^T I_{C_i} J_{\omega i}\right)\right]\dot{q}$$

Equation 29: Kinetic Energy of Total System

Equating Equation 29 to Equation 19 we find the Explicit form of Manipulator Mass Matrix, Equation 30.

$$M = \sum_{i=1}^{n}\left(m_i J_{vi}^T J_{vi} + J_{\omega i}^T I_{C_i} J_{\omega i}\right)$$

Equation 30: Explicit form of Manipulator Mass Matrix

15.1.3 Vector of centrifugal and Coriolis forces

We begin with Equation 23: Vector of centrifugal and Coriolis forces.

[141] Links here refers to the actuated limb segments of the exoskeleton correlating with the thigh, shin, and foot.

$$v(q,\dot{q}) = \dot{M}\dot{q} - \frac{1}{2}\begin{bmatrix} \dot{q}^T \frac{\partial M}{\partial q_1} \dot{q} \\ \vdots \\ \dot{q}^T \frac{\partial M}{\partial q_n} \dot{q} \end{bmatrix} = C(q)[\dot{q}^2] + B(q)[\dot{q}\dot{q}]$$

Sparing the derivation, we can say that givem Equation 31 and Equation 32, that Equation 33 and Equation 34 hold true.

$$m_{ijk} = \frac{\partial m_{ij}}{\partial q_k}$$

Equation 31: mijk

$$b_{ijk} = \frac{1}{2}(m_{ijk} + m_{ikj} - m_{jki})$$

Equation 32: Christoffel Symbols

$$C(q) = \begin{bmatrix} b_{111} & b_{122} & \dots & b_{1nn} \\ b_{211} & b_{222} & \dots & b_{2nn} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n11} & b_{n22} & \dots & b_{nnn} \end{bmatrix}$$

Equation 33: Coefficients associated with centrifugal forces

$$B(q) = \begin{bmatrix} 2b_{112} & 2b_{113} & \dots & 2b_{11n} & 2b_{123} & \dots & 2b_{12n} & \dots & 2b_{1(n-1)n} \\ 2b_{212} & 2b_{213} & \dots & 2b_{21n} & 2b_{223} & \dots & 2b_{22n} & \dots & 2b_{2(n-1)n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 2b_{n12} & 2b_{n13} & \dots & 2b_{n1n} & 2b_{n23} & \dots & 2b_{n2n} & \dots & 2b_{n(n-1)n} \end{bmatrix}$$

Equation 34: Coefficients associated with Coriolis force

Where

$$[\dot{q}^2]^T = [\dot{q}_1{}^2 \quad \dot{q}_2{}^2 \quad \dots \quad \dot{q}_n^2]$$

$$[\dot{q}\dot{q}] = [\dot{q}_1\dot{q}_2 \quad \dot{q}_1\dot{q}_3 \quad \dots \quad \dot{q}_1\dot{q}_n \quad \dot{q}_2\dot{q}_3 \quad \dot{q}_2\dot{q}_4 \quad \dots \quad \dot{q}_2\dot{q}_n \quad \dots \quad \dot{q}_{(n-1)}\dot{q}_n$$

15.1.4 Vector of gravity force

The vector of gravity force, g, represents the gravitational potential energy of the system. The gravitational potential energy of the system is given by the gravitational potential energy of every link in the system, see Equation 35.

$$V = \sum_{i=1}^{n} V_i$$

Equation 35: Potential Energy of the System

The gravitational potential energy of each link is given by Equation 36, where is the height of the centre of mass of the link relative to the origin (pelvis).

$$V_i = m_i g h$$

Equation 36: Gravitational Potential Energy of Each Link

Thus, we may say (using the Jacobian to map the location) the vector of gravity force, g, is given by Equation 37.

$$g = -(J_{v1}^T \quad J_{v2}^T \quad J_{v3}^T)\begin{pmatrix} m_1 g \\ m_2 g \\ m_3 g \end{pmatrix}$$

Equation 37: Vector of Gravity Force

16.Appendix F: Circuits

16.1 Voltage Follower

For a further reading or an introduction to the behaviour of operational amplifiers, op-amps, refer to The Art of Electronics (Horowitz & Hill, 2015). A voltage follower is an op-amp configuration where Equation 38 holds.

$$V_{in} = V_{out}$$

Equation 38: Voltage Follower

The topology of a voltage follower may be found in.

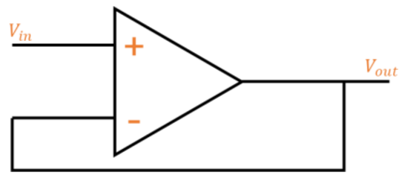


Figure 35: Voltage Follower

17.Appendix G: Assembled Rig

18.Appendix H: Code Snippets

18.1 Main – Apparatus A

```
while (1) {

        /* USER CODE END WHILE */

        /* USER CODE BEGIN 3 */
        Update_ADC_Values();

        // Update error (difference between values)
        // We want both sensors to be the same constant difference
        delta = (-ADC_D_Value + ADC_C_Value);
        set_p(par, delta);

        // Get control value
        update_control(par);
        delta = get_T_target(par);
        DC = DC + delta;

        // Update duty cycle
        set_pulse_width();

        /* Toggle LED */
        if (HAL_GetTick() > (epoch_LED + D_LED)) {
                HAL_GPIO_TogglePin(LD3_GPIO_Port, LD3_Pin);
                epoch_LED = HAL_GetTick();

        }

}
```

18.2 Main – Apparatus B

```
while (1) {

        /* USER CODE END WHILE */

        /* USER CODE BEGIN 3 */

        if (!sender) { // Receiver, this controller would manage the servo speed

                /* Read messages from UART 2 */
                HAL_UART_Receive_DMA(&huart2, RX_B2, len);

                /* Process messages */
                for (i = 0; i < B_SIZE; i++) {
                        // Read the message and save the location in the buffer
                        readMSG(msg, RX_B2, &x, &x0);

                        if (msg->complete) { // if a message is complete (valid)
                                RX_B2[x - 1] = '@'; // mark it as read
                                update_value(par, msg); // update system parameters

                                // Construct a message to be sent
                                contructMSG((char*) TX_B1, msg, B_SIZE);

                                // Update the duty cycle of the servo
```

```
            // (this was used in the demo, as the
            // other controller would tell this one the demand)
            DC = msg->value;
            set_pulse_width();

            // Tell the world the current DC
            memset(TX_B1, 0, B_SIZE);
            sprintf(TX_B1, "DC%u\tT%u\n\r", DC, get_T_target(par));
            while (isTransmitting(&huart1, &huart2))
                ;
            HAL_UART_Transmit_DMA(&huart1, TX_B1, B_SIZE);
            HAL_UART_Transmit_DMA(&huart2, TX_B1, B_SIZE);

        }
    }
}

if (sender) { // Transmitter, this controller would manage the load cell
    Update_ADC_Values();

    // translate load cell values for mass
    // (unlike the proximity sensors we
    // can't just make them equal and unscaled,
    // cause the load cells are .... bad)
    mA = -1.6614 * ADC_A_Value + 5178.8;
    mB = -17.484 * ADC_B_Value + 26220;

    // No negative loads
    mA = (mA < 0) ? 0 : mA;
    mB = (mB < 0) ? 0 : mB;

    // Add the value to the moving average buffer
    mA_Buff[loadIndex] = mA;
    mB_Buff[loadIndex] = mB;
    loadIndex = (loadIndex + 1) % loadBufferLen;

    // We add the most recent "loadBufferLen" number of measurements,
    // then divide them for a moving average
    loadBot = 0;
    loadTop = 0;
    for (i = 0; i < loadBufferLen; i++) {
        loadBot = loadBot + mA_Buff[i];
        loadTop = loadTop + mB_Buff[i];
    }

    // this is the dividing
    loadBot = loadBot / loadBufferLen;
    loadTop = loadTop / loadBufferLen;

    // We want to bias the top to run when the load is equal (by 500g)
    delta = (loadTop + 500);

    // If the bottom is greater than the top, we stop
    delta = (loadBot > (loadTop + 500)) ? 0 : delta;
    DC = delta;
    set_pulse_width();

    /* Create messages to send off */
    set_T_target(par, delta);
```

```
            contruct_X_msg('T', par, msgT, TX_T);

            /* Send out messages */
            while (isTransmitting(&huart1, &huart2))
                ;
            HAL_UART_Transmit_DMA(&huart1, (uint8_t*) TX_T, B_SIZE);
            HAL_UART_Transmit_DMA(&huart2, (uint8_t*) TX_T, B_SIZE);

        }

        /* Toggle LED */
        if (HAL_GetTick() > (epoch_LED + D_LED)) {
            HAL_GPIO_TogglePin(LD3_GPIO_Port, LD3_Pin);
            epoch_LED = HAL_GetTick();
        }

}
```

18.3 ADC Init

```
/* ADC1 init function */
static void MX_ADC1_Init(void) {

    ADC_MultiModeTypeDef multimode;

    /**Common config
     */
    hadc1.Instance = ADC1;
    hadc1.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;
    hadc1.Init.Resolution = ADC_RESOLUTION_12B;
    hadc1.Init.ScanConvMode = ADC_SCAN_DISABLE;
    hadc1.Init.ContinuousConvMode = DISABLE;
    hadc1.Init.DiscontinuousConvMode = DISABLE;
    hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
    hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;
    hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
    hadc1.Init.NbrOfConversion = 1;
    hadc1.Init.DMAContinuousRequests = DISABLE;
    hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
    hadc1.Init.LowPowerAutoWait = DISABLE;
    hadc1.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
    if (HAL_ADC_Init(&hadc1) != HAL_OK) {
        _Error_Handler(__FILE__, __LINE__);
    }

    /**Configure the ADC multi-mode
     */
    multimode.Mode = ADC_MODE_INDEPENDENT;
    if (HAL_ADCEx_MultiModeConfigChannel(&hadc1, &multimode) != HAL_OK) {
        _Error_Handler(__FILE__, __LINE__);
    }

    /**Configure Regular Channel
     */
    sConfig_A.Channel = ADC_CHANNEL_1;
    sConfig_A.Rank = ADC_REGULAR_RANK_1;
    sConfig_A.SingleDiff = ADC_SINGLE_ENDED;
    sConfig_A.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
    sConfig_A.OffsetNumber = ADC_OFFSET_NONE;
```

```
        sConfig_A.Offset = 0;
        if (HAL_ADC_ConfigChannel(&hadc1, &sConfig_A) != HAL_OK) {
                _Error_Handler(__FILE__, __LINE__);
        }

        sConfig_B.Channel = ADC_CHANNEL_2;
        sConfig_B.Rank = ADC_REGULAR_RANK_1;
        sConfig_B.SingleDiff = ADC_SINGLE_ENDED;
        sConfig_B.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
        sConfig_B.OffsetNumber = ADC_OFFSET_NONE;
        sConfig_B.Offset = 0;
        if (HAL_ADC_ConfigChannel(&hadc1, &sConfig_B) != HAL_OK) {
                _Error_Handler(__FILE__, __LINE__);
        }

}

/* ADC2 init function */
static void MX_ADC2_Init(void) {

        /**Common config
         */
        hadc2.Instance = ADC2;
        hadc2.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;
        hadc2.Init.Resolution = ADC_RESOLUTION_12B;
        hadc2.Init.ScanConvMode = ADC_SCAN_DISABLE;
        hadc2.Init.ContinuousConvMode = DISABLE;
        hadc2.Init.DiscontinuousConvMode = DISABLE;
        hadc2.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
        hadc2.Init.ExternalTrigConv = ADC_SOFTWARE_START;
        hadc2.Init.DataAlign = ADC_DATAALIGN_RIGHT;
        hadc2.Init.NbrOfConversion = 1;
        hadc2.Init.DMAContinuousRequests = DISABLE;
        hadc2.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
        hadc2.Init.LowPowerAutoWait = DISABLE;
        hadc2.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
        if (HAL_ADC_Init(&hadc2) != HAL_OK) {
                _Error_Handler(__FILE__, __LINE__);
        }

        /**Configure Regular Channel
         */
        sConfig_C.Channel = ADC_CHANNEL_1;
        sConfig_C.Rank = ADC_REGULAR_RANK_1;
        sConfig_C.SingleDiff = ADC_SINGLE_ENDED;
        sConfig_C.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
        sConfig_C.OffsetNumber = ADC_OFFSET_NONE;
        sConfig_C.Offset = 0;
        if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_C) != HAL_OK) {
                _Error_Handler(__FILE__, __LINE__);
        }

        sConfig_D.Channel = ADC_CHANNEL_2;
        sConfig_D.Rank = ADC_REGULAR_RANK_1;
        sConfig_D.SingleDiff = ADC_SINGLE_ENDED;
        sConfig_D.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
        sConfig_D.OffsetNumber = ADC_OFFSET_NONE;
        sConfig_D.Offset = 0;
        if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_D) != HAL_OK) {
```

```
        _Error_Handler(__FILE__, __LINE__);
    }

    sConfig_E.Channel = ADC_CHANNEL_3;
    sConfig_E.Rank = ADC_REGULAR_RANK_1;
    sConfig_E.SingleDiff = ADC_SINGLE_ENDED;
    sConfig_E.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
    sConfig_E.OffsetNumber = ADC_OFFSET_NONE;
    sConfig_E.Offset = 0;
    if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_E) != HAL_OK) {
        _Error_Handler(__FILE__, __LINE__);
    }

    sConfig_F.Channel = ADC_CHANNEL_4;
    sConfig_F.Rank = ADC_REGULAR_RANK_1;
    sConfig_F.SingleDiff = ADC_SINGLE_ENDED;
    sConfig_F.SamplingTime = ADC_SAMPLETIME_1CYCLE_5;
    sConfig_F.OffsetNumber = ADC_OFFSET_NONE;
    sConfig_F.Offset = 0;
    if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_F) != HAL_OK) {
        _Error_Handler(__FILE__, __LINE__);
    }

}
```

18.4 UART Init

```
/* USART1 init function */
static void MX_USART1_UART_Init(void) {

    huart1.Instance = USART1;
    huart1.Init.BaudRate = 9600;
    huart1.Init.WordLength = UART_WORDLENGTH_8B;
    huart1.Init.StopBits = UART_STOPBITS_1;
    huart1.Init.Parity = UART_PARITY_NONE;
    huart1.Init.Mode = UART_MODE_TX_RX;
    huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
    huart1.Init.OverSampling = UART_OVERSAMPLING_16;
    huart1.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
    huart1.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
    if (HAL_UART_Init(&huart1) != HAL_OK) {
        _Error_Handler(__FILE__, __LINE__);
    }

}

/* USART2 init function */
static void MX_USART2_UART_Init(void) {

    huart2.Instance = USART2;
    huart2.Init.BaudRate = 9600;
    huart2.Init.WordLength = UART_WORDLENGTH_8B;
    huart2.Init.StopBits = UART_STOPBITS_1;
    huart2.Init.Parity = UART_PARITY_NONE;
    huart2.Init.Mode = UART_MODE_TX_RX;
    huart2.Init.HwFlowCtl = UART_HWCONTROL_NONE;
    huart2.Init.OverSampling = UART_OVERSAMPLING_16;
    huart2.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
    huart2.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
```

```
	if (HAL_UART_Init(&huart2) != HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}

}
```

18.5 PWM Init

```
/* TIM3 init function */
static void MX_TIM3_Init(void) {
	/* Counter Prescaler value */
	uint32_t uhPrescalerValue = 0;

	/* Compute the prescaler value to have TIM3 counter
	/* clock equal to 1000000 Hz */
	uhPrescalerValue = (uint32_t) ((SystemCoreClock / 2) / 1000000) - 1;

	TIM_ClockConfigTypeDef sClockSourceConfig;
	TIM_MasterConfigTypeDef sMasterConfig;
	TIM_OC_InitTypeDef sConfigOC;

	htim3.Instance = TIM3;
	htim3.Init.Prescaler = uhPrescalerValue;
	printf("Prescaler Value: %lu\r\n", uhPrescalerValue);
	htim3.Init.Period = PWM_PERIOD;
	printf("Period Value: %lu\r\n", PWM_PERIOD);
	htim3.Init.ClockDivision = 0;
	htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
	htim3.Init.RepetitionCounter = 0;
	htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
	if (HAL_TIM_Base_Init(&htim3) != HAL_OK) {
		printf("HAL_TIM_Base_Init failed.");
		_Error_Handler(__FILE__, __LINE__);
	}

	sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
	if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}

	if (HAL_TIM_PWM_Init(&htim3) != HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}

	sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
	sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
	if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig)
			!= HAL_OK) {
		_Error_Handler(__FILE__, __LINE__);
	}

	// sConfigOC.Pulse = 0;
	sConfigOC.OCMode = TIM_OCMODE_PWM1;
	sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
	sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
	sConfigOC.OCNPolarity = TIM_OCNPOLARITY_HIGH;
	sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;

	sConfigOC.Pulse = (uint32_t) (PWM_PERIOD / 2);
	if (HAL_TIM_PWM_ConfigChannel(&htim3, &sConfigOC, TIM_CHANNEL_1)
			!= HAL_OK) {
		/* Configuration Error */
		Error_Handler();
	}

	HAL_TIM_MspPostInit(&htim3);

}
```

18.6 Update_ADC_Values

```
/* USER CODE BEGIN 4 */
/**
 * @brief  Update all the ADC sensor readings (global variables)
 * @note   We stop and start the ADCs here as STM is buggy and its
 * best not to function as expected. As in all things, if it doesn't work,
 * turn it on and off again
 */
void Update_ADC_Values(void) {
	/* Read ADC_A
	 * ADC A */
	if (HAL_ADC_ConfigChannel(&hadc1, &sConfig_A) != HAL_OK) {
		Error_Handler();
	}
	if (HAL_ADC_Start(&hadc1) != HAL_OK) {
		Error_Handler();
	}
	if (HAL_ADC_PollForConversion(&hadc1, 1) == HAL_OK) {
		ADC_A_Value = HAL_ADC_GetValue(&hadc1);
	}
	if (HAL_ADC_Stop(&hadc1) != HAL_OK) {
		Error_Handler();
	}
	HAL_Delay(1);
	/* Read ADC_C
	 * ADC B */
	if (HAL_ADC_ConfigChannel(&hadc1, &sConfig_B) != HAL_OK) {
		Error_Handler();
	}
	if (HAL_ADC_Start(&hadc1) != HAL_OK) {
		Error_Handler();
	}
	if (HAL_ADC_PollForConversion(&hadc1, 1) == HAL_OK) {
		ADC_B_Value = HAL_ADC_GetValue(&hadc1);
	}
	if (HAL_ADC_Stop(&hadc1) != HAL_OK) {
		Error_Handler();
	}
	HAL_Delay(1);

	/* Read ADC_C
	 * ADC C = PA4 = A3 = ADC2 Channel 1 */
	if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_C) != HAL_OK) {
		Error_Handler();
	}
	if (HAL_ADC_Start(&hadc2) != HAL_OK) {
		Error_Handler();
	}
```

```
        if (HAL_ADC_PollForConversion(&hadc2, 1) == HAL_OK) {
                ADC_C_Value = HAL_ADC_GetValue(&hadc2);
        }
        if (HAL_ADC_Stop(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        HAL_Delay(1);

        /* Read ADC_D
         * ADC D = PA5 = A4 = ADC2 Channel 2 */
        if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_D) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_Start(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_PollForConversion(&hadc2, 1) == HAL_OK) {
                ADC_D_Value = HAL_ADC_GetValue(&hadc2);
        }
        if (HAL_ADC_Stop(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        HAL_Delay(1);

        /* Read ADC_E
         * ADC E = PA6 = A5 = ADC2 Channel 3 */
        if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_E) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_Start(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_PollForConversion(&hadc2, 1) == HAL_OK) {
                ADC_E_Value = HAL_ADC_GetValue(&hadc2);
        }
        if (HAL_ADC_Stop(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        HAL_Delay(1);

        /* Read ADC_F
         * ADC F = PA7 = A6 = ADC2 Channel 4 */
        if (HAL_ADC_ConfigChannel(&hadc2, &sConfig_F) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_Start(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        if (HAL_ADC_PollForConversion(&hadc2, 1) == HAL_OK) {
                ADC_F_Value = HAL_ADC_GetValue(&hadc2);
        }
        if (HAL_ADC_Stop(&hadc2) != HAL_OK) {
                Error_Handler();
        }
        HAL_Delay(1);

        return;
}
```

18.7 Set Duty Cycle

```c
/**
 * @brief    Sets the Duty cycle (pulse width) of the servo motor
 * @note     The (max,min) and (maxL,minL) parameters refer to the
 * two servomotors in use (MG995 and 900-8 respectively)
 *  @note    The DC is global variable, so we have no inputs or outputs
 */
void set_pulse_width(void) {
	// Ensure value is valid
	DC = (DC > 4096) ? 4096 : DC;
	DC = (DC < 0) ? 0 : DC;

	// init
	int DutyCycle = (DC < 0) ? -DC : DC; // ensure always positive
	DutyCycle = 4096 - DutyCycle;
	int max = 2500, min = 700;
	int maxL = 1700, minL = 1500;
	int servoMotor = 1;
	int littleServoMotor = ! servoMotor;

	// For the MG995
	if (servoMotor) {
		HI_PERIOD = (((max - min) * DutyCycle / 4096.0)) + min;
		HI_PERIOD = 2 * HI_PERIOD;
	}

	// For the 900-8

	if (littleServoMotor) {
		HI_PERIOD = (((maxL - minL) * DC / 4096.0)) + minL;
		HI_PERIOD = 2 * HI_PERIOD;
	}

	// This was used for debugging and calibration
	int zero = 0;
	if (zero) {
		HI_PERIOD = 2000 * (1.3);
	}

	// Once we have the value, we set the DC
	__HAL_TIM_SET_COMPARE(&htim3, TIM_CHANNEL_1, HI_PERIOD);
}
```

18.8 PID

```c
/**
 * @brief    Updates the control value, this is the output of K(s)
 * and is our command to the actuators to achieve error == 0
 * @param    par : system paramaters
 */
void update_control(struct PARAMETERS* par) {

	// get e
	e = get_p_target(par) - get_p(par);

	// update moving average for intergral term
	maaPush(par->q, e);
	// Error for proportional term
	Ep = e;
	// Error for integral term
	Ei = get_integral(par->q);
	// Error for derivatve term
	Ed = e - par->e_last;

	// Set the target torque
	set_T_target(par, (par->Kp * Ep + par->Ki * Ei + par->Kd * Ed));
	par->e_last = e;
	return;
}
```

18.9 Is Transmitting

```c
/**
 * @brief    A safety to ensure we don't access the DMA buffer while transmitting
 * @param  huart1 : UART 1 Handle
 * @param  huart2 : UART 2 Handle
 * @retval !0 = transmitting, 0 == not busy
 */
int isTransmitting(UART_HandleTypeDef *huart1, UART_HandleTypeDef *huart2) {
	return ((huart1->gState != HAL_UART_STATE_READY)
			|| (huart2->gState != HAL_UART_STATE_READY)) ? 1 : 0;
}
```

19.Appendix I: Mechanical Design

19.1 Mount Structure

It could not be presumed that the exoskeleton segments would have free ends, so the system would need to be designed to be attached, firmly, to a rod of an arbitrary shaped cross section of an arbitrary size, without access to a free end. It was required that wobbling vibration, or movement of any kind was to be minimised and the connection could be removed and reattached an indeterminant number of times.

Hose clamps were identified as a suitable fastener method. Screw/band (worm gear) clamps are reusable, can be applied to a rod of an arbitrary shape and size (with ranges), affix firmly, and may be attached quickly with a screwdriver. However, as the details of the proposed exoskeleton became available it was noted that the cross section of exoskeleton frame may have been as small as 5mm in diameter[142]. A size below the range of standard hose clamps. Instead, cable ties were identified as an ideal fastener method.

Cable ties, or zip ties are a form of typically plastic ratcheting strap. The can be affixed to a rod of an arbitrary shape and size, attached by hand, and are disposable. While a less permanent solution for an attachment mechanism compared to hose clamps, the were deemed sufficient for a proof of concept.

To mount the measurement structure to the exoskeleton a plat was design that could sit flush to the frame. As seen in Figure 36: Mount Structure, the structure (black), could be mounted to the exoskeleton frame A. Seen from the side, gutters where placed (B) so cable ties could be affixed, while guard rails (C) ensured the cable ties did not slip or move during operation. The measurement structure and any auxiliary objects could be affixed at the surface of the plate (D) with counterbored sections (E) for nuts and bolts to be mounted while sitting flush with the surface of the exoskeleton.

[142] A full scale exoskeleton (not a proof of concept designed to hold no load) would presumable have thicker links)

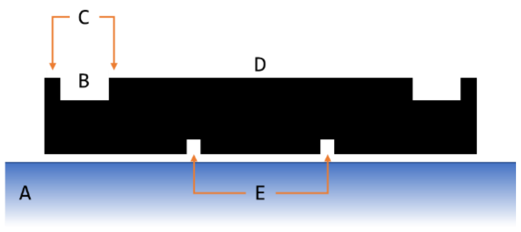


Figure 36: Mount Structure

The component was created in Autodesk Inventor, as seen in “\CAD Schematics” or the attached documents. As discussed in 3.2.7 - Controller Mount, the mount structures for the ankles required that two controller boards be mounted on the same structure, a double mount structure was also designed, see “\CAD Schematics” or the attached documents.