Gait Control of a Lower Limb Exoskeleton for Strength Augmentative Applications

A thesis submitted in partial fulfilment of the requirements for

the award of the degree of

B. Tech

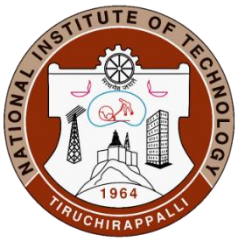
in

Mechanical Engineering

By

ADITHYA VENKATA NARAYANAN (111120001)

R CHARAN BHARDHWAJ (111120109)



DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

TIRUCHIRAPPALLI – 620015

May 2024

Bonafide Certificate

This is to certify that the project titled "**Gait Control of a Lower Limb Exoskeleton for Strength Augmentative Applications**" is a bonafide record of the work done by

Adithya Venkata Narayanan (111120001)

R Charan Bhardhwaj (111120109)

in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in **MECHANICAL ENGINEERING** at the **NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI**, during the year 2023-2024.

Dr. N Siva Shanmugam	**Dr. D Ezhilarasi**	**Dr. K Pannirselvam**
Project Guide	Project Co-Guide	Head of the Department

Project Viva-voce held on: _____

Internal Examiner **External Examiner**

ABSTRACT

Lower body exoskeletons have emerged as promising tools for augmenting human strength and mobility, offering significant potential for enhancing physical performance and assisting individuals with mobility impairments. The research arena of lower-limb exoskeletons focuses mainly on 12 DOF (Degrees of Freedom) models, and the market prices of these models are unaffordable since a significant chunk of the cost are attributed to the actuators.

The physical labour force exerts themselves to the maximum capabilities to carry heavy weights in work site This can lead to workers injuring themselves in the long run. Strength augmenting exoskeletons can be the solution to this problem, if made more affordable. This thesis aims to investigate and analyse the capabilities of an 8 DOF (a reduced DOF) lower limb exoskeleton to augment the strength of the user in performing lifting activities.

To analyse the capabilities of the exoskeleton in augmenting user strength, the human gait cycle was executed by making the setup autonomous. Maxon motors were configured and tuned as per requirement as the motor drivers for Maxon motors can execute a low-level current and speed control. Once the motors were configured and the test bed setup was complete, the angular orientation was collected as the feedback using an inertial measurement unit (IMU) which was used to calculate the difference between the error in angular position of the exoskeleton's links. A proportional controller was used to compute the angular velocity input, to be fed to the motor controller, and thus actuating the motor to reach the set point. A reference gait, which is the set of thigh and shin angles to be traced was taken and fed to the microcontroller, providing necessary PWM signals to make the links trace the desired gait trajectory.

Keywords: Lower Limb Exoskeleton, Gait, Walking Trajectory, Speed Control, Motor Control, Hybrid Control Architecture.

ACKNOWLEDGEMENTS

We would like to extend our sincere gratitude to Dr. G. Aghila, Director of National Institute of Technology, Tiruchirappalli, for the invaluable cooperation extended to us, which enabled the successful completion of our project.

We are deeply obliged to Dr. K. Pannirselvam, the Head of the Department of Mechanical Engineering, for his unwavering support and encouragement. We are also grateful to the esteemed members of the evaluation committee, Dr. S Suresh, Dr. N. Siva Shanmugam, Dr. Bishweshwar Babu, and Dr. Raghu Ram Desu, for their valuable advice, which significantly contributed to the refinement of our project. We would like to sincerely thank the department of mechanical engineering for facilitating and allowing us to pioneer and work on this thesis.

We owe a profound debt of gratitude to our guide, Dr. N Siva Shanmugam, for his understanding, exceptional guidance and constant encouragement while facing highs and lows throughout the duration of our research work. His expertise played a vital role in our research journey, expanding our horizons and we are truly privileged to have had the opportunity to work under his supervision.

We would also like to express our heartfelt gratitude to Dr. D Ezhilarasi (Co-guide), for her continuous support. Her expertise and inspiration played an important role in our journey, and we are truly blessed to have worked under her supervision as our co-guide.

Finally, we would like to acknowledge and thank our parents, relatives, and friends for their unwavering love, emotional support, and unwavering belief in our abilities, which provided us with the strength to overcome challenges and achieve our goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

1. DOF – Degrees of Freedom
2. BLDC – Brushless DC Motor
3. PD – Proportional Derivative
4. DC – Direct Current
5. ESC – Electronic Speed Controller
6. MOSFET – Metal Oxide Semiconductor Field Effect Transistor
7. PWM – Pulse Width Modulation
8. EMF – Electromotive Force
9. UART – Universal Asynchronous Receiver Transmitter
10. I2C – Inter Integrated Communication
11. TX – Transmitter
12. RX – Receiver
13. SDA – Serial Data
14. SCL – Serial Clock
15. IMU – Inertial Measurement Unit
16. FSR – Force Sensitive Resistor
17. EMG – Electromyography
18. EEG – Electroencephalogram
19. FOS – Factor of Safety
20. AWG – American Wire Gauge

NOMENCLATURE

1. θ – Angular Position
2. ω – Angular Velocity
3. α – Angular Acceleration
4. τ - Torque
5. i - Current
6. V - Voltage
7. l - Length
8. m – Mass
9. F - Force
10. I – Polar Moment of Inertia
11. g – Gravity
12. K_P – Proportional Gain
13. K_D – Derivative Gain

CHAPTER 1

INTRODUCTION

1.1 Background

Human exoskeletons robots are wearable robots that are powered by means of pneumatic, hydraulic and, motors and gearbox or cable-driven (electromechanical actuators), for multiple purposes. These include exoskeletons for assistive applications, for rehabilitation exercises and for strength augmentative applications.

1.1.1 Brief History of Exoskeletons

N. Yagn's conceptual model of robotic skeleton lays the foundation for the research of exoskeleton. He designed a lower extremity enhancement model that was shaped like a bow operating in parallel to the legs of the user as depicted in the Fig 1.1. Later in 1965, General Electric designed the first full body powered exoskeleton that had 30 DOF and weighed 680 Kgs (Makinson, 1971). The design is shown in the Fig 1.2.

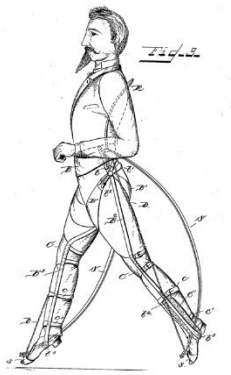


Figure 1.1: N. Yagn's Exoskeleton design. (US Patent No. 420179, 1890)

Figure 1.2: Hardiman 1 Full Body Exoskeleton

The Human Engineering and Robotics Laboratory (HERL), University of California, Berkeley, designed and developed the first functional energetically autonomous load

carrying human exoskeleton, the “Berkeley Lower Extremity Exoskeleton,” the BLEEX in the year 2004. BLEEX was capable of walking at a pace of 1.3m/s under no load condition and was able to walk at a pace of 0.9 m/s when a payload of up to 75 kg

Figure 1.3: BLEEX (H. Kazerooni, 2006)

was loaded (H. Kazerooni, 2006) (Vikash Kumar, 2019).

1.1.2 Types of Exoskeletons

Exoskeletons, based on their applications are primarily categorized into three types, assistive, rehabilitation and augmentative exoskeletons.

Exoskeletons for assistive applications focuses on individuals who are unable to walk normally due injuries that deal with the spinal cord, stroke or susceptible to effects of aging. These exoskeletons are usually controlled by employing pre-defined trajectories which are triggered by detecting the user’s intentions.

Exoskeletons are also used for the purpose of rehabilitation of patients who are recovering from a surgery as well. Exoskeletons in these categories are designed to be mounted to treadmill and their portability is not a priority when designing. Their control primarily focuses on ensuring that the amount assistance provided decreases as time passes if the patients using these rehabilitative robots develop more strength.

The third category of exoskeletons are designed to augment the strength and ability of a healthy human. These types of exoskeletons are primarily targeted in the military applications to allow the personnel to be able lift heavy objects such as weaponry or

other combatants in the event of an emergency. These types of exoskeletons should be capable of complimenting the work done by the user. (Hao Lee, 2019).

In an augmentative exoskeleton, the assistance/strength augmentation can be directly applied to the tasks. Such design of exoskeletons is not flexible and cannot be adapted to different environments. On the other hand, is a class of augmentative exoskeletons where strength assistance is applied on the user. This class of exoskeletons are weight and power restricted as they cannot exceed weight limits of 150kg because it will reduce the agility of the user completely (M. Fontana, 2014).

1.2 Motivation

A study done by Dasari Karthik et. al focusing on the masonry labour productivity in building construction sites in India concluded that physical fatigue caused due to extreme working conditions and lifting heavy weights accounts for 81% in Relative Importance Index (RII). Another study conducted by D. A. Neumann proved that severe injuries can be incurred through lifting heavy weights especially in regions such as the lower back.

The common physical labour workforce is involved in lifting heavy weights daily that can adversely affect their joint, muscle and their spinal health. Significant measures must be taken to ensure that this workforce is less affected by adverse lifting conditions. The motivation behind this thesis work is to provide cost-effective strength augmentative lower limb exoskeleton that can be used in extreme situations to carry loads that can affect human body.

1.3 Objectives

After a thorough analysis of the factors that have to be considered making the exoskeleton cost-effective, the following objectives were shortlisted for this thesis.

The objectives are as follows:

a) To configure torque controller and design position controller for a BLDC motor to reach and hold the position.
b) To deploy a PD + Bang Bang controller to trace a desired gait pattern on both external weight-loaded and without external weight-loaded conditions.
c) To analyze the differences in gait patterns and study contact forces at foot with and without load when using an exoskeleton and when not using an exoskeleton.

1.4 Chapter Overview

This section will provide an overview of the chapters. In chapter 2, an extensive literature was conducted on topic of exoskeleton design considerations, control strategies deployed in exoskeleton are presented. Chapter 3 will elucidate all the important terminologies and the concepts needed to understand this thesis. Chapter 4 elaborates the experimental setup and chapter 5 is dedicated to the research methodology undertaken to address the chosen statement. Chapter 6 provides the conclusion and summarizes the result, and the future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Impacts of lifting heavy weights

Kinoshita et. al investigated the change in the biomedical parameters of the human body when the subjects walked under normal conditions and under load carrying conditions. 10 healthy male individuals were taken as test subjects and were made to carry load in two different configurations and analysed the impact it had on the human musculoskeletal structure. The research concluded that with increasing load, the gait, and the body posture of the individual changes significantly irrespective of the load lifting configuration. It was also concluded that the chance of encountering injuries due to stress was also possible.

Another study by Hayder F N Al Shukla et. al discusses how improper manipulation of heavy weights can lead to experiencing hard stresses throughout the human body. The lower spine is commonly considered the most susceptible region to be affected by lifting heavy weights. The paper also discusses the biomechanical modelling and the multi-level control strategies that must be deployed to reduce the force wrench and achieve better trajectory. The research work concludes that for an exoskeleton system, the user intention has to be identified, minimize the human-machine interaction, estimate the interaction force wrench for control purpose, and finally determine the power compensation that the exoskeletal system has to provide to compensate for the lacking effort.

2.2 Gait study

Martin et. al worked extensively in the field of gait analysis have done significant and impactful contributions in the field. The work complied gait data for people of different age groups performing activities. The paper presented the angular positions, angular velocities, and acceleration data values to name a few for the hip joint, knee joint and ankle joint. Collection of this data allows the quick visualization of the gait pattern and deployment of control strategies on an exoskeleton.

Humans' gait patterns are based off dynamic balanced gait and on the other exoskeletons are based off statically balanced gait. The difference between human and

exoskeleton gait will give path to interesting modelling of the gait of the exoskeleton. (Barbareschi G, 2015).

2.3 Design considerations for Exoskeleton

2.3.1 Actuator

Lower Limb exoskeletons are actuated predominantly in three types, i.e. active, passive, and quasi passive assistive actuation, and hybrid actuation.

Exoskeletons that are actuated with powered components such as motors, hydraulics or pneumatics are considered active actuation. When passive elements such as springs are used that stores energy, the stored energy can be used to actuate the system. Such systems are referred to as passive actuation systems. These systems use gravity to achieve balance. Quasi passive actuation systems work in conjunction with viscosity devices like clutch-damper configurations (Sai K. Banala, 2006). Hybrid systems work in a combination of active passive, or active quasi passive system (Conor James Walsh, 2007).

2.3.2 Degree of Freedom

The complexity of the lower limb exoskeleton system can be attributed to the DOF of the system. The DOF of a lower limb exoskeleton is defined as the number of independent ways a joint can move. The maximum degree of freedom for a human leg is considered 7. The thigh has 3 DOF, knee has 1 DOF and ankle joint consists of 3 DOF summing up to 7 degrees of freedom. (Zhiguo Feng, 2007).

2.4 Control Strategies

Exoskeletons are complex systems that demand multi-level control strategies for the smooth operation. Generally, there are three levels of hierarchical controllers. These controllers are namely the high-level, mid-level and low-level controllers (Norazam Aliman, 2017). The highest-level controller determines the intention of the human being and calculates the joint angles corresponding to that intention of motion. Once these angles are computed, these values are passed down to the mid-level controller that send pulses to the low-level controller that is directly linked to the actuator. The low-level controller ensures that the actuator reaches the desired final set point. With this overall structure, different control strategies are used to determine the intention, the pulses to be sent to the low-level controller (Michael R Tucker, 2015).

CHAPTER 3

BASIC CONCEPTS AND REPRESENTATION

In this chapter some of the basic notions and terms associated with the area of exoskeleton will be presented. All the terminologies that will be used throughout in this thesis will be covered. This allows a common understanding and lucid understanding of the terms that are being used.

3.1 Representations

It is necessary to define how the coordinates are represented. The global frame is a right-handed coordinate system with the directions of the axis as shown in the figure. The coordinate axis is aligned such that the X-axis is aligned in the walking direction. The analysis of an exoskeleton is easier in the orthogonal planes and are planes are named the following:

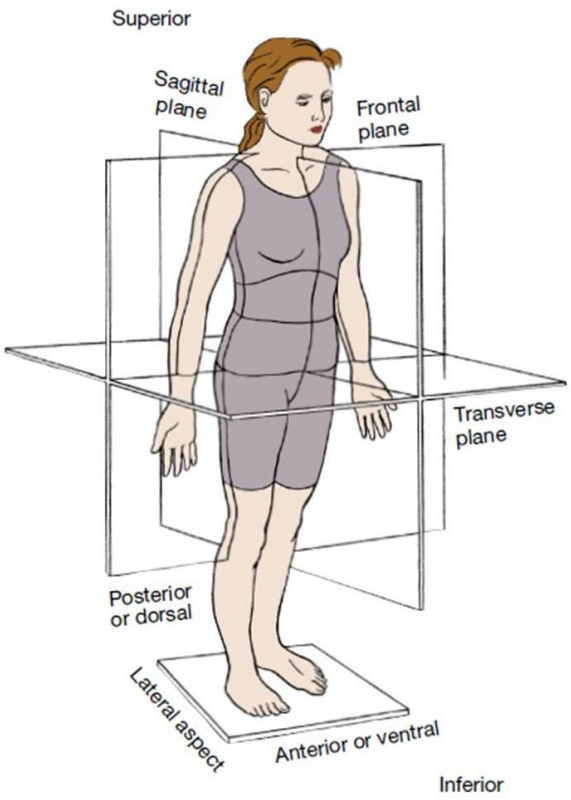


Figure 3.1: Planes of analysis.

XY plane – Transverse plane

XZ plane – Sagittal plane

YZ plane – Frontal plane

3.2 Gait Terminology

3.2.1 Robotic Locomation

This section is mainly based on (M. Vukobratovic, 2006) who have made great contribution towards the unification and standardization in robotic locomotion. The terms that are related to robotic locomotion are:

Walk: Defined as movement by putting forward each foot into turn and not moving both feet of the ground at once.

Gait: Defined as the manner of walking hence when an exoskeleton is used to walk it can have several different gates. If a vector θ(t) is defined to contain all the joint angles, then a time history of θ(t) represents a specific gait.

Step: A step is defined as in: the direction of motion during the contact with the ground the leg from the front position with respect to the trunk comes to the rear position then it is deployed from the ground and in the transfer phase move to the front position to make again contact with the ground and the cycle is repeated.

Periodic Gait: A gait is periodic if the same step is repeated after a fixed interval of time, thus if θ(t) = θ(t+T) implies that the gait is preriodic with period T (one step duration).

Symmetric Gait: A gait is symmteric if the step can be divided into two equal time-periods such that the right leg behaves as left leg in the other period and vice versa. Thus if $\theta_R(t) = \theta_L(t + T/2)$, where θ_R is the joint angles of the right leg and θ_L is the joint angles of the left leg, the gait is said to be symmetric.

Trajectory: A trajectory is the path of a moving point in the three dimensional space. In gait analysis, trajectories are used to describe the movement of a joint frame, relative to the global reference frame, which is usually atttached to the ground.

3.2.2 Walk Cycle

The walk cycle as described in (Walter Pirker, 2016) consists of the following important phases, the swing phase and the stance phase.

Stance phase: It constitutes approximately 60% of the gait cycle. Both feet are in contact with the ground at the start and the end of the stance phase. It can be further subdivided into the following:

i. Heel Strike – initial contact
ii. Mid – stance
iii. Terminal Stance

Swing phase: It consititutes the other 40% percent of the gait cycle and it can be subdivided into the following:

i. Toe – off – initial swing phase
ii. Mid – swing
iii. Terminal swing

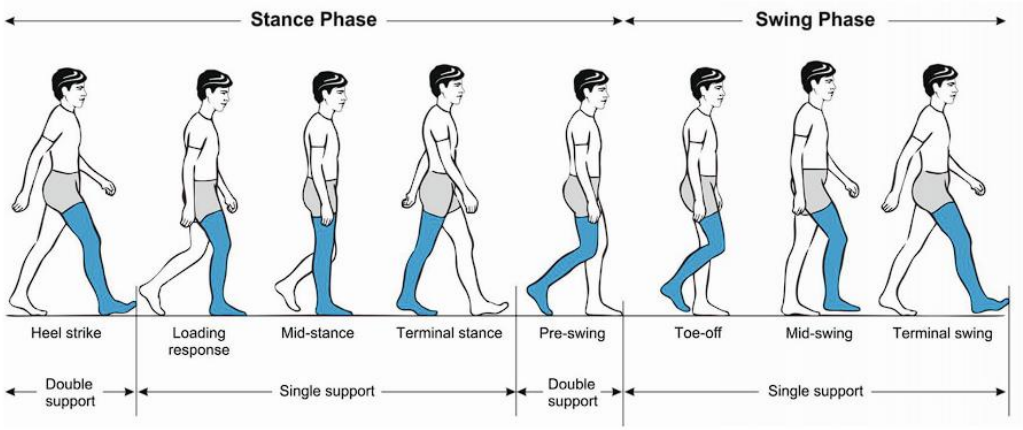


Figure 3.2: Phases of walking cycle. Source - *(Walter Pirker, 2016)*

3.3 Brushless DC Motor and its Controller

3.3.1 Advantages of choosing Brushless DC Motor

Exoskeletons can be actuated using brushless DC motor as they have the following characteristics.

1) BLDC motors have inherently higher torque than their counterparts in similar voltage range without an additional transmission system.
2) They are back-drivable and can be suitably damped electronically to mimic human joint behaviour.

3) BLDC motors usually have hall sensors inside the motor housing that trigger output when the rotor passes over them in specific position. This inbuilt addition allows for deployment of better control architecture deployment.
4) BLDC motor have electrical commutator as compared to mechanical commutator in brushed DC motor. This gives the following advantages.
    a. Longer lifetime because of reduced wear and tear.
    b. Minimized losses due to frictional losses and heat losses.
    c. Higher switching frequency which allows for lower noise.

3.3.2 Control of BLDC Motor

3.3.2.1 ESC

Electronic Speed Controllers (ESCs) regulate the speed of Brushless DC (BLDC) motors by adjusting the voltage provided to the motor's coils. from the MOSFETs by changing electrical signals at a constant frequency.

ESCs use PWM signals from a microcontroller or receiver to control motor speed. The PWM signal's duty cycle (ratio of ON time to the total period) determines the average voltage supplied to the motor. BLDC motors have three coils, and the ESC sends signals to energize each coil in a sequence, creating a rotating magnetic field. By varying the timing and duration of these signals, the ESC controls the motor's speed and direction.

Advanced ESCs incorporate feedback mechanisms such as Hall Effect sensors or back electromotive force (EMF) sensing to precisely regulate motor speed and provide features like smooth acceleration, braking, and even reverse operation.

3.3.2.2 Torque Control on a BLDC Motor

In a BLDC motor, the current (directly corresponds to the torque output) is typically controlled through a closed-loop control system employing PWM techniques. This control system utilizes feedback from current sensors, such as shunt resistors or Hall effect sensors, to monitor the current flowing through the motor windings. The control algorithm calculates the difference between the desired current reference and the actual measured current, generating an error signal. This error signal is then used to adjust the duty cycle of the PWM signals applied to ESC. By modulating the duty cycle of the PWM signals, the controller regulates the average voltage applied to the motor windings, thereby controlling the current flow.

The torque of a BLDC motor is given by the equation (Sul, 1995):

$$\tau_{elec} = \frac{e_a i_a + e_b i_b + e_c i_c}{\omega_m}$$

Where, e_a, e_b, e_c are Back EMF's and i_a, i_b, i_c are currents in each phase.

3.4 Communication Protocols

In a complex robotic system such as an exoskeleton, high processing power is mandatory. In the event of lack of processing power, multiple microcontrollers are used to decentralize the control flow and distribute processing power. On such occasions, it is of prime importance that the compute units can communicate with one another effectively. Communication protocols such as UART and I2C are used primarily in these circumstances.

3.4.1 Universal Asynchronous Receiver and Transmitter (UART) Protocol

UART is the simplest form of serial communication. It has a transmitter and receiver line that send and receive bits respectively (Durda, 2021). It is called an asynchronous line as there is no clock signal line that time syncs the data sent/received. The bits are sent based on the "baud rate," the frequency at which the bits are sent in that channel. A baud rate of 9600 implies that the microcontroller will read data on the communication line at a frequency 9600 Hz. The timing of reading, once set, is executed based on the internal crystal oscillator pulse timings. Depending on the number of communication lines, UART is classified into 3 types:

A. Simplex – This configuration has only one data line and the sending or receiving bits happens in the same line. The data can be sent only in one direction.

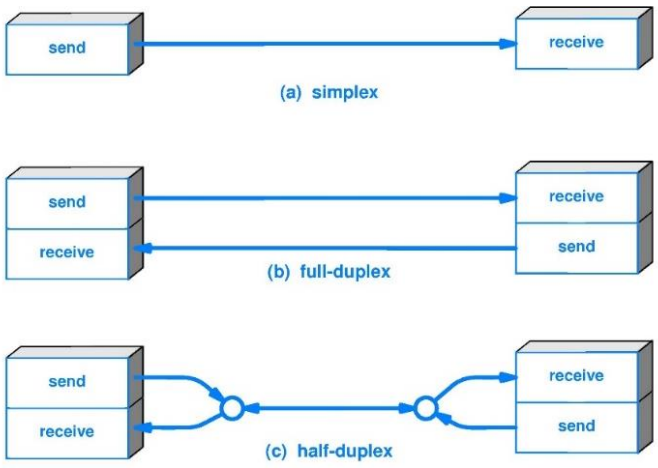


Figure 3.3: Configurations of UART communication.

B. Half Duplex – There is a single line for transmitting and receiving respectively but one of the two microcontrollers can use the communication line at a given point in time.
C. Full Duplex – There are two communication lines TX and RX and simultaneously sending and receiving across these lines are possible.

3.4.2 I2C Protocol

The Inter-Integrated Circuit (I2C) protocol is a serial communication standard used to connect microcontrollers and peripherals, enabling bidirectional data transfer over a two-wire interface. Devices are distinguished by unique addresses, facilitating communication between multiple components on the same bus. Communication occurs via a master-slave architecture, where a master device initiates data transfer and controls the bus, while slave devices respond to commands. There are two lines, the SDA (the data line) and SCL (the clock line). The communication protocol is considered synchronous as there exists clock line that maintains the synchronism. The data on the data line from each bit is read against the rising and falling of the clock line pulses.

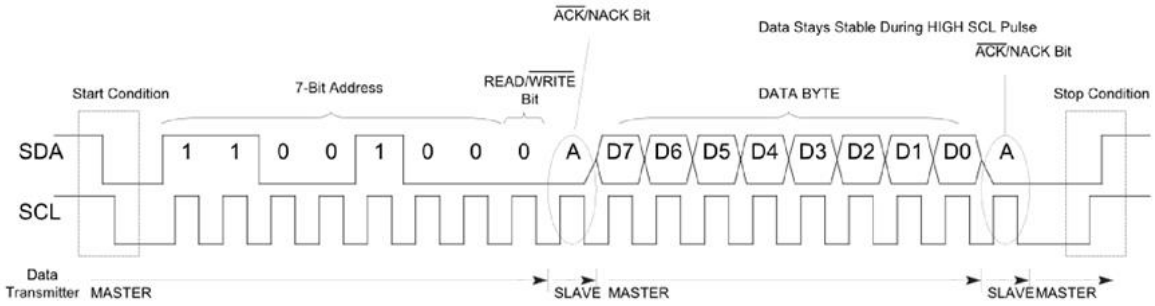


Figure 3.4: I2C Line Configuration

CHAPTER 4

EXPERIMENTAL SETUP

4.1 Elements required for an Exoskeleton System

The following aspects are needed in an experimental prototype of an exoskeletal system:

4.1.1 The Actuators

As previously discussed, using an active exoskeleton augments the user's strength. As such, the actuators in any exoskeleton play an extremely crucial role in the functioning and will be chosen with extreme consideration.

Table 4.1: Comparison of Pneumatic/Hydraulic actuators vs Motors

Pneumatic/Hydraulic Actuators	Motors
Typically, the motion caused by them is linear.	The motion caused by them, transmitted from the shaft is rotational.
Joint angles are changed by arranging in a 3-bar mechanism with a link having variable length.	Joint angles are changed by actuating the motor, in turn causing rotary motion directly.
The gas/oil in the system tends to leak out thus dirtying workspace and needing replenishment.	They need batteries to be powered. They do not have any consumable/perishable elements in construction.
They are robust in nature and can withstand impacts from objects and environment.	They are delicate and cannot withstand impact from surroundings, especially with encoders or other sensitive elements.
They are easier to maintain and service and are cheaper.	They are more expensive to service and repair. The initial investment is lower.

4.1.2 The Sensor Suite

For proper control and functioning of an exoskeleton, the sensor systems become the core in obtaining and sending feedback based on which the system's actions will be decided. The following are the kinds of feedback needed, the corresponding sensors for the same, typically used in an exoskeleton:

a. Angular position
    i. Rotary encoders (at joints)
    ii. Potentiometers (at joints)
    iii. Accelerometers/IMUs (on links)
b. Contact Forces (typically at locations like finger pads and under the sole)
    i. Force Sensitive Resistors (FSRs)
    ii. Strain Gauge
c. Neuro-muscular signals
    i. EMG (muscle activity detection)
    ii. EEG (cerebral activity)

Angular positions (and higher order terms - velocity and acceleration) are taken as feedback for control algorithms to close the loop. At the same time, combinations of inputs from contact forces and neuro-muscular signals will yield appropriate activity/intention detection and, as such, are used in action planning.

4.1.3 Computational Unit

The feedback from the sensor suite must be processed into meaningful outputs and used to provide the inputs corresponding to the system's necessary actions. This raises the requirement of a unit that can be interfaced with both the sensors and actuators while performing the necessary computations to plan the action sequence of the system and transmit the same to the actuators. For this task, a microcontroller is used as it is sufficiently low level to communicate with peripherals while also being able to perform minimal necessary computation.

4.1.4 The Frame

The frame is the collection of links and other entities that hold all the sensors and actuators together. The frame also consists of all the load-bearing entities (the links connecting the joints) and the harness that connects the wearer to the links (the lumbar support for the spine, the pads for the thigh, and the shank).

The lumbar support is a vital element in the frame, where the wearer is hung, and the exoskeleton is grounded to the human. This supporting link also serves as the start in the chain of links that serve as an alternate load path for the user's weight to the ground. As such, the support and the harness should be designed to cover a broader area to

decrease the pressure on the wearer at the regions of contact as it bears the user's weight when hung.

4.2 Calculations and Considerations

4.2.1 The Frame

The links connecting the motor will be loaded with a bending moment during the entire range of motion. As such, they must be able to withstand the bending load. The design must be optimized to be minimal to ensure that the actuator can support the structure and the user in the desired manner.

Most of the frame elements are made from aluminum, and the interfaces to the motors are made of Delrin, ensuring that the structure has low weight and that the motors require lower torques.

Assumptions for calculations of bending moment:

i. The links are treated as cantilever beams, hanging from one end (from the motor end).
ii. Loads acting on them are not continuous and are point loads. The location of point load acting is the centre of mass of the system of elements considered.
iii. Self-weight of the links has been excluded from the calculation as they are close to 2% of the weight of the user's body part at the same location.
iv. For calculation of bending load, the links are assumed to be in a horizontal position so that the bending load is maximized on the system.
v. The acceleration due to gravity is taken as 9.8144 m/s^2, and the Elastic Modulus (E) of Aluminum as 70 GPa.

Bending Moment Calculations (ExRx, n.d.) (Robert V. Schulte, 2023)

i. Shank:

   Dimensions of shank link: 400mm x 50.8mm x 0.6mm

   Mass of Shank (m_{shank}) = 3.336kg

   Mass of foot (m_{foot}) = 0.971kg

   Distance of the centre of mass of shank from point of rotation (l_{shank}) = 0.21m

   Distance of the centre of mass of foot from point of rotation (l_{foot}) = 0.45m

   Distance of the centre of mass of foot and shank (combined) from point of rotation:

$$l_{com} = \frac{(m_{shank} * l_{shank}) + (m_{foot} * l_{foot})}{m_{shank} + m_{foot}}$$

l_{com} = 0.264m

Load (F) = m_{com}*g = 42.27 N

Polar moment of inertia (I):

$$I = \frac{l * b * (l^2 + b^2)}{12} m^4$$

I = 16090*10^{-9} m^4

The maximum deflection in the beam due to bending load δ_{max}, for a load acting at x from supported end:

$$\delta_{max} = \frac{F * x^2 * (3L - x)}{6EI}$$

From above equation, values and with x = l_{com}, δ_{max} = 0.0408 µm

ii. Thigh:

Dimensions of thigh link: 400mm x 50.8mm x 0.6mm

Mass of Thigh (m_{thigh}) = 10.563kg

Mass of Motor (m_{motor}) = 1.4kg

Mass of Shank (m_{shank}) = 3.336kg

Mass of foot (m_{foot}) = 0.971kg

Distance of the centre of mass of thigh from point of rotation (l_{thigh}) = 0.206m

Distance of the centre of mass of motor from point of rotation (l_{motor}) = 0.4m

Distance of the centre of mass of shank & foot from point of rotation (l_{shank}) = 0.667m

Distance of the acting centre of mass from point of rotation:

$$l_{com} = \frac{(m_{thigh} * l_{thigh}) + (m_{motor} * l_{motor}) + (m_{com\_shank} * l_{com\_shank})}{m_{thigh} + m_{motor} + m_{com\_shank}}$$

l_{com} = 0.3444m

Load (F) = m_{com}*g = 159.41 N

Polar moment of inertia (I):

$$I = \frac{l * b * (l^2 + b^2)}{12} mm^4$$

I = 6817.5*10^{-8} mm^4

The maximum deflection in the beam due to bending load δ_{max}, for a load acting at x from supported end:

$$\delta_{max} = \frac{F * x^2 * (3L - x)}{6EI}$$

From above equation, values and with x = l_{com}, δ_{max} = 0.29 µm

From calculations, it is observed that the deflection in the links on full loading is in the order of micrometres, which is negligible. As such, it is concluded that the setup is safe to test, load and execute experiments on without leading to failure.

4.2.2 Motors

The significant factors that help decide the actuator specification for the system are the joint torques and speed, i.e., the motor's power. For the exoskeleton to trace the necessary gait, it must be able to move loads at necessary speeds to achieve needed positions. For this set of calculations, the static torque is calculated for holding a specific pose, and it is multiplied by a factor of safety to account for its functioning in dynamic conditions.

$$\tau_{knee} = \frac{(m_{shank}) * g * l_{shank} * \sin\theta_{knee}}{2}$$

$$\tau_{hip} = \left[\left(m_{motor} + \frac{m_{thigh}}{2}\right) * g * l_{thigh} * \sin\theta_{hip}\right] + \left[m_{shank} * g * \left(l_{thigh} * \sin\theta_{thigh} + \frac{l_{shank} * \sin\theta_{knee}}{2}\right)\right]$$

The parameters are the same as considered for the bending moment calculations, with the addition of angles. The maximum condition for thigh is both thigh and knee angle being at 37°, while for shank it is 45°.

From above equations and parameters, the τ_{knee} = 5.17Nm, and τ_{hip} = 30.14Nm. Considering a factor of safety of 2 and a speed of 10 rpm at the described position, the maximum power needed will be:

$$P_{max} = \tau_{hip} * \omega * FOS$$

P_{max} = 60.28W

Nominal Motor Power Output from datasheet = 197.88W. From this, it can be conclusively said that the motors can handle human loading within all working angle ranges required to demonstrate human gait.

4.3 Setup Description

The experimental setup used consists of the following:

4.3.1 Frame

The frame has links made of aluminum, and the mating structures to the motors made of Delrin were designed to handle the bending and shear loads acting on them sufficiently.

4.3.2 Motor and Controller

The motors used are Maxon EC-φ90 flat BLDC motors with hall sensors and built-in encoders. The specs of the motor are as follows:

1. Working Voltage: 48V
2. Nominal Torque: 964 mNm
3. Nominal Current: 4.06 A
4. Gearbox Knee: 36:1
5. Gearbox Thigh: 60:1
6. Encoder Resolution: 4096 pulses per revolution

The motor, along with the motor controller ESCON Module 50/8, can operate with up to 15A of current. The motor controller can operate with a two-channel double-ended encoder input, hall sensor inputs from the motor, and an internal controller to perform accurate current and speed control on the motor [Appendix A].

4.3.4 Wiring

All the power lines are wires of 16 AWG, which can comfortably conduct a current of 13A without overheating. Furthermore, the data cables are wires of 22AWG, which can lower the losses during conduction.

Switches of appropriate ratings have been chosen on each of the power lines: 20A for the motor and 6A for the buck converter (which powers the compute element).

4.3.5 Sensors

Four 9-DOF IMUs, BNO-055, are used as sensors to close the control loop. The IMU, chosen for its sensor outputs, can give acceleration, gyroscope, magnetometer, and other readings [Appendix B] over I2C to a master controller. Gravity readings are processed to obtain the information needed for motor control.

4.3.6 Compute Element

An ESP32 Devkit, based on Espressif's ESP32 series microcontroller, is used as the main compute element. It was chosen based on clock speed, support with peripherals, PWM, I2C, UART, and Digital I/O capabilities.

A PCB has been designed to minimize the exposed ends and to turn the system into a singular embedded autonomous module. [Appendix C]

CHAPTER 5

RESEARCH METHODOLOGY

5.1 Control System Design

5.1.1 Current Control Strategy

The current traveling across a shunt resistor on an excited phase is measured and compared against the setpoint value. If lower, the motor coils are excited with a higher voltage. The coils' excitation voltage is lowered if the current is higher than the setpoint. This simple change in current translates to a more complicated kinematic change from the standpoint of position control.

The electromagnetic torque of a BLDC motor is given by the equation (Sul, 1995):

$$\tau_{elec} = \frac{e_a i_a + e_b i_b + e_c i_c}{\omega_m}$$

Where, e_a, e_b, e_c are Back EMF's and i_a, i_b, i_c are currents in each phase. At any given time, one of the three windings will be given the positive voltage and one negative, which completes the circuit. Since the same amount of current (with no accumulation of charge at any point, by Kirchoff's Current Law) passes through to complete the circuit, it can be extrapolated that the phase currents between the running phases will be equal. The sum of back emf between the two phases will amount to the voltage difference, V, provided by excitation. Thus, the relation evolves as follows:

$$\tau_{elec} = \frac{V * i}{\omega_{shaft}}$$

For a simple system of a link fixed at a point, free to rotate in the vertical plane, the relation between the mechanical torque and the angular position (θ, with respect to vertical) is:

$$\tau_{mech} = m_{link} * g * l_{link} * \sin\theta$$

During the functioning of the motor, the electromagnetic torque output is equal in magnitude to the demand mechanical torque. Similarly, the shaft angular velocity is the same as the angular velocity of the link. From the above two torque equations, the relation between current and angular position is:

$$i = \frac{m_{link} * g * l_{link} * sin\,\theta * \dot{\theta}}{V}$$

Combining all the constant terms into an arbitrary term, λ, the current required for the link to be at a specific angular position θ and having angular velocity $\dot{\theta}$ is given by:

$$i = \lambda * sin\,\theta * \dot{\theta}$$

In practice, the equation would not end as shown above due to non-linearities or noise accumulated in the response through the whole system's functioning. As such, the equation would be:

$$i = \lambda * sin\,\theta * \dot{\theta} \; + \; b$$

Where, λ is a constant close to $(m_{link} * g * l_{link})$ but not equal, and an additional constant bias term *b* is introduced to account for offsets. Different input signals must be given to the motor to observe the response of the system at various angular positions and time instances to be able to accurately model the system equation.

Accounting for the inrush current factor (for the start of the actuation of the motor) and penalizing error of angular position and velocity. A modified version of PD controller with model information can be used:

$$i = f\left(\theta, \dot{\theta}\right) + \left(K_p * \Delta\theta\right) + \left(K_d * \left(\frac{d(\Delta\theta)}{dt} - \omega_\theta\right)\right)$$

This control equation can be interpreted as:

i. When desired angular velocity and position are reached, the other terms tend to 0. As such, the system is supplied the current of value $f\left(\theta, \dot{\theta}\right)$, corresponding to the current necessary to be at angular position θ, with the angular velocity $\dot{\theta}$.
ii. The $\left(K_p * \Delta\theta\right)$ term, is the proportional term from the proposed addition of PD controller, serves two distinct and important purposes. This helps in the system reaching the setpoint position quickly and as an inrush current factor (for the motor to begin actuation).

iii. The $\left(K_d * \left(\frac{d(\Delta\theta)}{dt} - \omega_\theta\right)\right)$ term, is the derivative term, also serves two purposes. It acts as a higher order proportional term (for angular velocity), and acts as a dampening factor for oscillations against angular position.

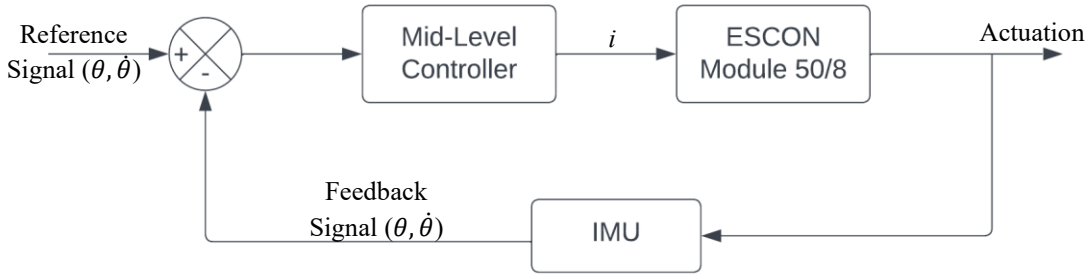


Figure 5.1: Current Control

The above control equation is obtained by treating the system as a second order system, and while using only zero and first order inputs. Additionally, second order inputs can be added to the equation, resulting in a control scheme that can minimize fluctuations in angular velocity while also allowing to achieve desired angular acceleration.

5.1.2 Speed Control

Traditionally, speed control is done using feedback from sensors such as optical incremental, absolute encoders, or hall-effect sensors. In this, a voltage is given for coil excitation while measuring the speed of the motor shaft against the setpoint value. If lower, the motor coils are excited with a higher voltage. The coils' excitation voltage is lowered if the speed exceeds the set point. This scheme is more straightforward to execute for the position control task since the change is directly reflected in the form of a kinematic entity that is also being controlled.

Since the kinematic entity is being controlled, angular velocity ($\dot{\theta}$) is an order of magnitude higher than the target variable (θ). So, a simple PD control scheme can perform the necessary position control task. As such, the control equation will be:

$$\omega = (K_p * \Delta\theta) + \left(K_d * \frac{d(\Delta\theta)}{dt}\right)$$

5.1.3 Control Scheme

The goal of achieving a set target position can be achieved through the above mentioned two control modes (configurable and usable via ESCON Module 50/8 motor controller). As such, a study of formulation of the execution method was done to conclude that a speed-based position controller is optimal for deployment at the beginning stage of testing and research.

The position controller used in practice for the thesis does not have a derivative term since the ESCON Module's internal speed controller [Appendix A] can accurately achieve setpoint angular velocities, and the controller response (slow or stiff) is configurable. Thus, the control equation is:

$$\omega = (K_p * \Delta\theta)$$

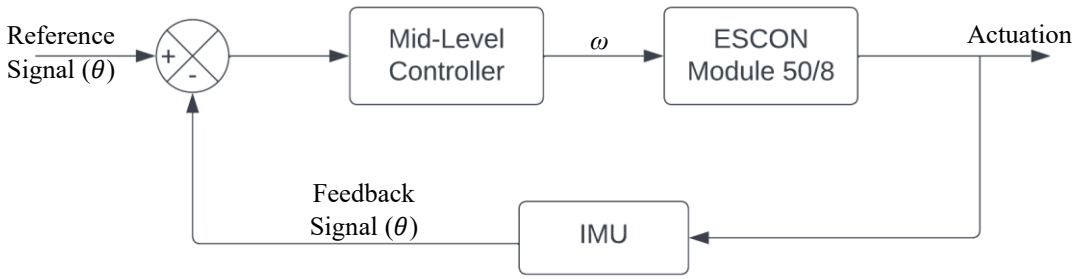


Figure 5.2: Speed Controller

5.1.4 System Inputs and Outputs

From the above controller equation, the following information can be gathered:

i. The controlled quantity is the angular velocity of the shaft.
ii. The target quantity is the angular position of the shaft, and the error is computed based on the deviation from the setpoint.

From the above information, the inputs to the controller are:

i. Reference Signal: This is the setpoint, or the target position, to be achieved.
ii. Angular Position: This is the real-time angular position of the shaft in the vertical plane, based on which the error is computed.

The controller output will be the angular velocity value for the motor to achieve the desired position based on the configured response characteristic.

5.1.5 System Constraints

The system and controller are operated under the following constraints:

i. Maximum allowed angle for the thigh during the gait cycles is 25° from the vertical.
ii. Maximum allowed angle for the shank during the gait cycles is 37° from the vertical.
iii. The maximum allowed angular velocity for any link is 10 rpm. As such, the maximum permissible speed for hip and knee motors are 600 and 360 rpm, respectively.
iv. Maximum permissible current draw per motor is limited to 15A.
v. The control frequency is restricted to 20Hz, for the initial stages.
vi. The maximum angle difference for a link between any two consecutive setpoints should be no more than 15°.

5.2 Elements of the control system

From the above control block diagram representations, the following are functional elements on which computation is performed.

5.2.1 Feedback

Feedback is an essential component of a closed-loop control architecture, as the subsequent future output of the system is based on its current state. As mentioned in Section 4.3.5, BNO-055 was used as the feedback element for the position controller. The IMU can provide gravity outputs on its X, Y, and Z axes. The value of the angle from the vertical is obtained by using the following relation:

$$\theta = tan^{-1}\frac{g_x}{g_y}$$

5.2.2 Controller

The error is computed by comparing the feedback against the setpoint value. Then, it is multiplied by a proportional gain (K_P) to get the controller output. This output is mapped to an integer of 102 to 922 (10-90% duty cycle, on 10-bit resolution) to be given as input to the ESCON Module 50/8.

The higher the K_P value, the higher the speed at which the angular position is approached. This value determines the response speed of the controller. Very low K_P value means that the system takes a very long time to reach the setpoint while a very large setpoint means that the rate of change of angular position and velocity will be very high. This will cause unwanted jerks and disturbances in the system and is to be avoided. An ideal method to tune the value is to use the method of averages against the metric being desired reaching and settling time from one setpoint to another.

5.3 Controller Tuning

The position controller design, tuning, and implementation were done in stages to allow sufficient time for system behaviour analysis during deployment, to avoid accidental damage to equipment. The deployment on the system has been in the following order:

5.3.1 Single Actuator System

The controller design began with verifying checks of angular position feedback robustness. This was performed throughout the working range of the shin link, and outliers were handled to ensure consistent and reliable outputs were given to the controller.

The knee motor was set to operate on its nominal current 4.06A, as the datasheet provided. The speed was limited to 108 rpm during the testing phase, resulting in 3 rpm on the output shaft. The internal current and speed (closed and open loop) controllers were tuned using the auto-tuning package provided by the ESCON Studio application, with inputs for direction and speed control configured on both the receiving motor controller end and pins set for output and duty cycling on the MCU.

The knee motor was mounted on a table with various setpoint angles given while changing the gain on the controller. This activity was performed iteratively following the method of averages to reach the ideal K_P value for the desired settling time.

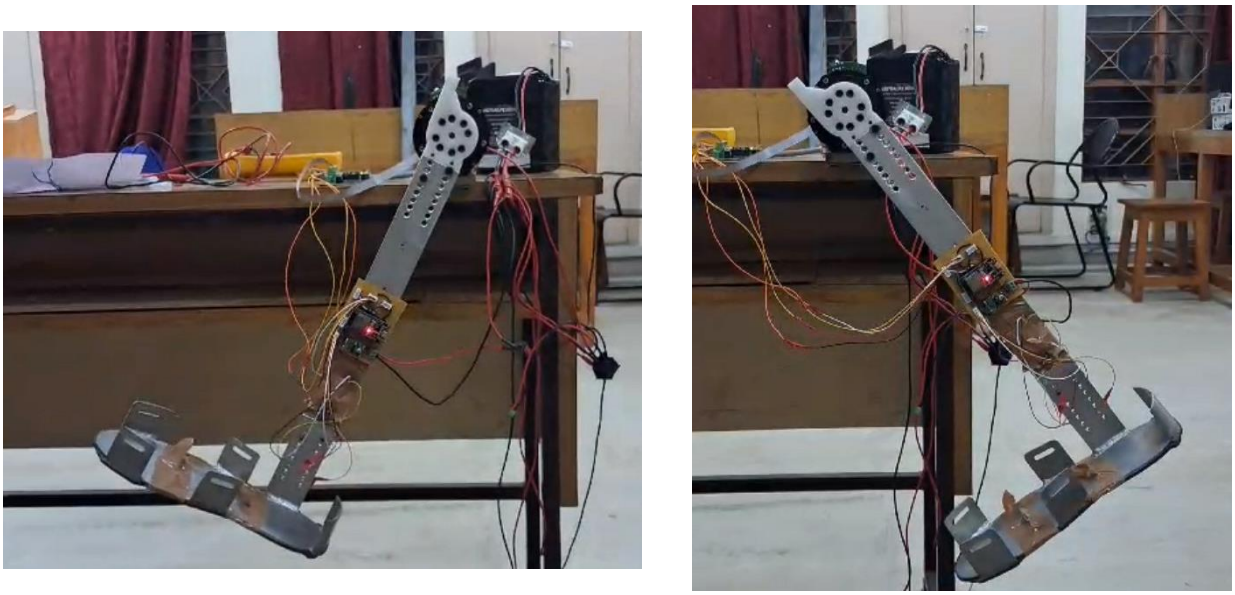


Figure 5.3: Shin link position control

5.3.2 Dual Actuator System – Single Leg Motion

The robustness check of the data stream from the IMUs was redone, with checks for crosstalk between the data on the bus. Stress tests were done to check the highest sampling and compute the IMU and MCU combination frequency against the desired control frequency to isolate the lagging element. The processed output data stream had an ODR of over 100Hz, which proved sufficient to deploy with controllers. As such, the control algorithm was deployed.

The hip motor was set to operate on a nominal current of 8.00A, as the joint needed to output higher nominal torque than the knee joint. The speed was limited to 120 rpm for the testing, resulting in 2 rpm on the output shaft. The internal current and speed (closed and open loop) controllers were tuned using inputs and pins for direction and speed control configured on both the receiving motor controller end and the MCU.

In this case, all the load was unmounted from the shaft of the hip motor during the auto-tuning sequence as it could cause unwanted noise during the software's internal current

demand and speed actual sampling. High-frequency angular vibration, caused as part of the auto-tuning routine, on objects (the links and attached motors) with heavy inertia will draw dramatically higher currents than needed and could potentially damage the gears on the transmission line.

The knee motor was mounted back on the hanging exoskeleton setup to the end of the thigh link. Specific setpoint angles were given while changing the gain on the controller till the ideal K_P value was reached. Post controller tuning, angles of specific stances were fed as input to validate that the overall system was working.

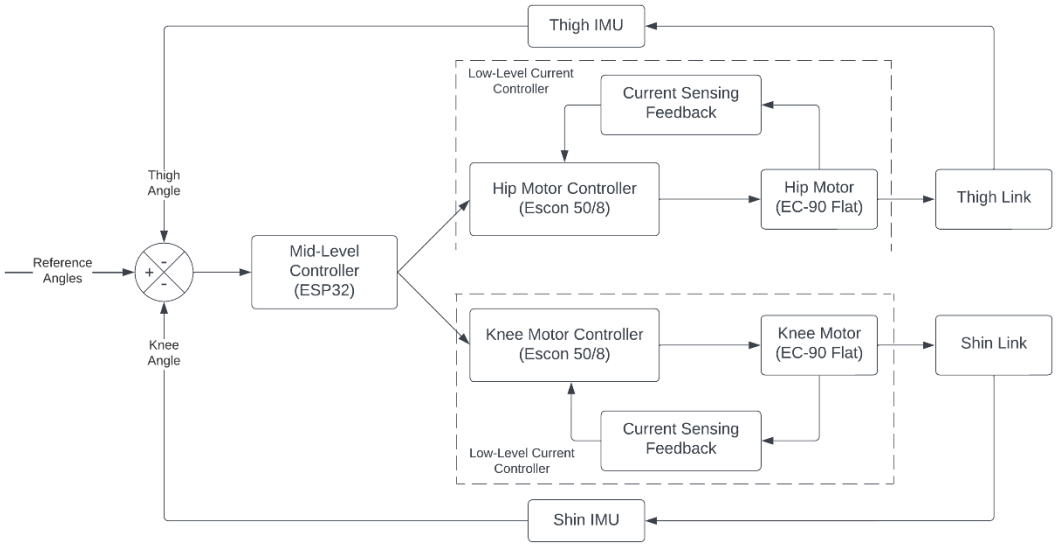
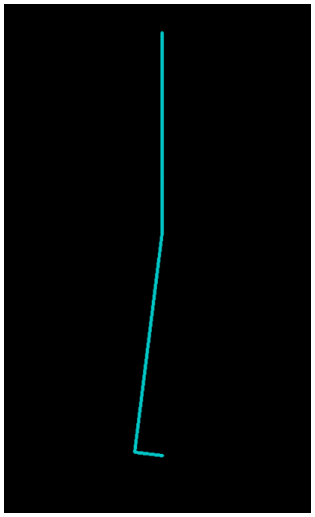


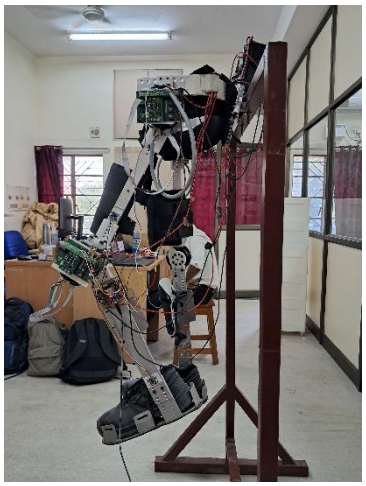
Figure 5.4: Overall control architecture.

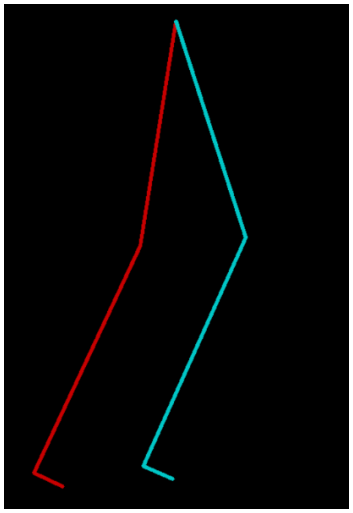
5.4 Gait Pattern Data

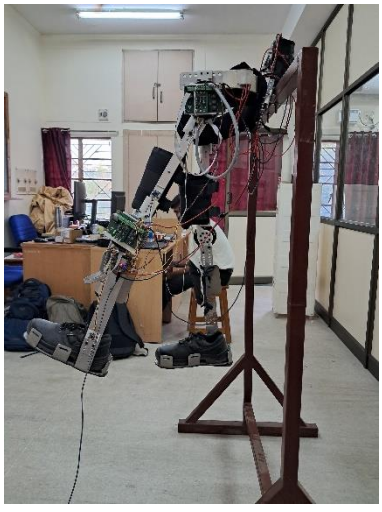
Schulte et al., 2023 generated a database detailing the kinematics of the lower extremity during gait related actions in able-bodies people. The gait data of one individual from the given sample set was taken, for the implementation of the thesis work.

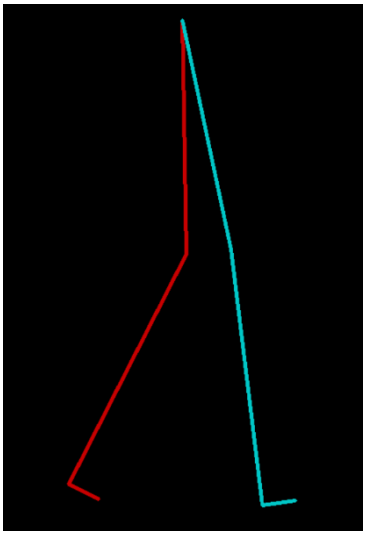
The data points of each subject had outputs of angular position, velocity, acceleration, muscular and electrical activity, joint torques of the hip, knee, and ankle joints at a frequency of 1000Hz collected for a time interval of 70 seconds.











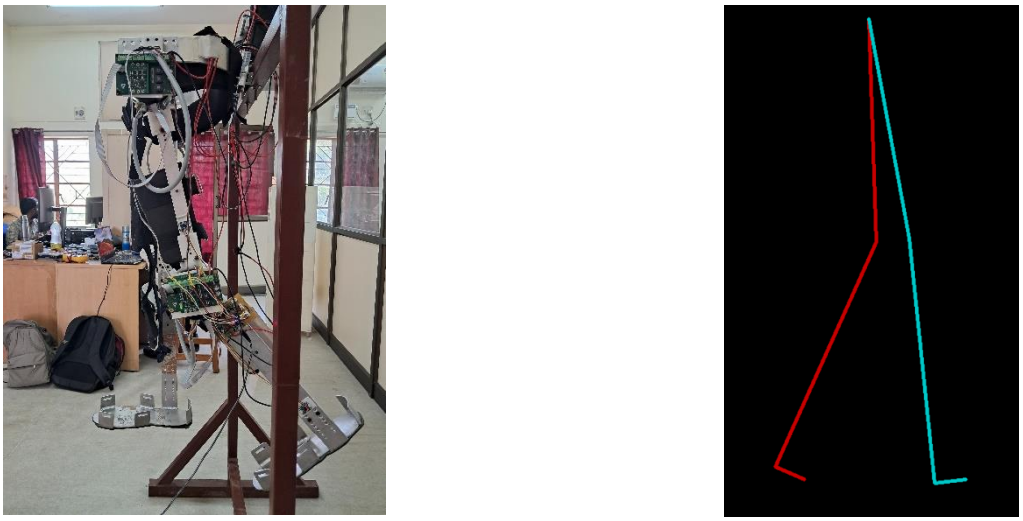


Figure 5.5: 2 DOF single leg gait cycle

5.4.1 Isolation of Single Step Motion Data Points

The graphs of the angular position data of the thighs and shanks were visualized to isolate the sequence of angles of a single step taken by the subject. The data was subsequently processed to isolate a set of 21 angles from the original 1000 data points, ensuring no significant losses in graph definition. The following are the plots of the 21 angles of each limb against time:

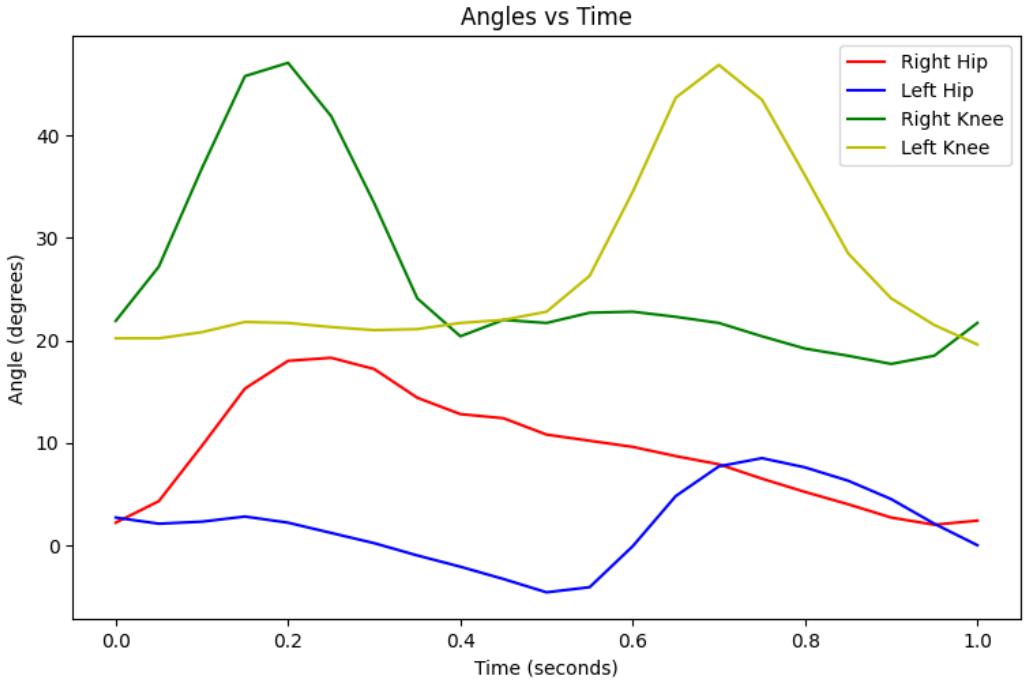


Figure 5.6: Single gait cycle plot

5.4.2 Dataset Augmentation and Visualization

The data points collected by Schulte et al., 2023 were that of humans with normal flexibility and no specific motor function restrictions. This data cannot be translated as is into the exoskeleton's working due to constraints imposed by the design of the setup. The knee joint has a guard tab that restricts that motion to prevent damage to the user's knees and not rotate into an angular position that is not humanly achievable. This guard tab also imposes an additional angular constraint that the minimum difference between the angles of the links should be 9 degrees. Hence, the dataset was augmented with this condition being the front-runner and shifting from the coordinate frame from the one Schulte et al., 2023 operated on to which the exoskeleton is calibrated. This angle sequence was also visualized using a Python script [Appendix F]

5.5 Continuous Gait Motion

Implementing a continuous gait trajectory means that the links must attain and move through the specific trajectory point sequence. As such, it is crucial to understand and define what it means to reach a position to the controller since the system's response is transient. Furthermore, all the work discussed in the previous sections was executed on a single leg. Time synchronization between the motion of the two legs is of paramount importance as asynchronous motion will not result in a replication of human gait but would cause the user to fall.

5.5.1 Position Attainment Condition

Since the working of the system is not completely predictable, placing a trigger within a region of tolerance from the target angle to act as a check measure of pose attainment is not the right approach. Experiments were conducted where a specific set of angles were fed as setpoints, and the system's response was observed. It was noted that the system reaches the setpoints within roughly 20% of the settling time but would fall out of the tolerance region at roughly about 50% of the settling time. This is because the thigh link moves because of the change in the angle of the shank and, subsequently, the torque required. So, at a particular instant, the torque demand would not match the supply. This causes an undershoot or overshoot of the position. To avoid the transient response conditions, the system's settling time is observed for a few

repetitions of the experiment, and a timer is used as the metric to determine the change in setpoints.

Hip Swing Up: 91-105

Trial. No.	Start Time			First Setpoint Reach			End of First Sighting			Final Stabilization			Time to reach point first time (ms)	Coasting Time (ms)	Instability Time (ms)	Settling Time (ms)
	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec				
1	1	31	625	1	34	789	1	36	182	1	40	52	3164	1393	3870	8427
2	1	57	272	2	0	578	2	1	875	2	6	915	3306	1297	5040	9643
3	2	19	814	2	23	128	2	24	757	2	27	928	3314	1629	3171	8114
Average Times													3261	1439	4027	8728

Shin Swing Up: 79-75

Trial. No.	Start Time			Final Stabilization			Settling Time (ms)
	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	
1	1	31	625	1	38	308	6683
2	1	57	272	2	5	514	8242
3	2	19	814	2	27	605	7791
Average Times							7572

Hip Swing Up: 91-105

Trial. No.	Start Time			First Setpoint Reach			End of First Sighting			Final Stabilization			Time to reach point first time (ms)	Coasting Time (ms)	Instability Time (ms)	Settling Time (ms)
	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec				
1	1	43	908	1	47	227	1	48	671	1	53	320	3319	1444	4649	9412
2	2	10	734	2	14	93	2	15	815	2	18	561	3359	1722	2746	7827
3	2	29	366	2	32	539	2	33	934	2	38	689	3173	1395	4755	9323
Average Times													3283	1520	4050	8854

Shin Swing Up: 79-75

Trial. No.	Start Time			Final Stabilization			Settling Time (ms)
	Minutes	Seconds	Millisec	Minutes	Seconds	Millisec	
1	1	43	908	1	52	62	8154
2	2	10	734	2	18	470	7736
3	2	29	366	2	37	431	8065
Average Times							7985

Table 5.1: System Behaviour w.r.t time

5.5.2 Time Synchronization

The mid-level controllers are executed using the ESP32D as the core processing element. Both legs' ESP32s work on the same internal clock frequency of 80MHz. As such, time synchronization of their action execution can be crudely achieved by having both processors begin operating at the same point in time. For this, one of the MCUs can exit its set up and be put to wait indefinitely for an acknowledgment handshake from the other MCU. The sender MCU will be transmitting the handshake and subsequently begin working with no delay. The receiver, on receiving, immediately begins working. A transmission delay, if it occurs, can be digitally compensated on the sender end to ensure synchronicity.

CHAPTER 6

RESULTS, CONCLUSION AND FUTURE WORK

6.1 Results and Conclusion

The following are the results achieved:

i. Low-level current and speed controllers have been configured for the BLDC motor control.
ii. A novel model-based current control architecture has been theorized, and the method of deployment has been elaborated.
iii. Mid-level position control based on a Proportional Bang-Bang controller has been deployed and tested.
iv. System was studied and an open loop feedforward controller has been deployed and verified for changing the setpoints to achieve continuous motion.
v. Time Synchronization in working of multiple controller units has been ideated, executed, and validated.

The following conclusions can be drawn from the research:

i. Repeating gait cycles and achieving poses is task that should be handled in parallel for multiple DOF systems. Microcontrollers with more than one core should be employed for the task of position control, with each core monitoring one motion.
ii. A speed-based control, although easier to deploy, is not the most robust solution to the position control task. This is due to the absence of an element of impedance control or a direct torque-based input. As such, it is advisable to work on current based control approaches.

6.2 Future Work

Developments that can be made to the existing test bed involve robust control architecture based on current control of motors can lead to precise control over the variables such as angular acceleration, position, and velocity. A sensor suite of IMUs can be used to detect the intention of the user and determine the joint angles for the gait

appropriately. Precise intention detection is possible with the help of EMG sensors that give data on the muscle activity.

CHAPTER 7

REFERENCES

Al-Shuka, H. &. (2019). Biomechanics, actuation, and multi-level control strategies of power-augmentation lower extremity exoskeletons: an overview. *International Journal of Dynamics and Control*, 1462-1488.

Barbareschi G, R. R. (2015). Statically vs dynamically balanced gait: Analysis of a robotic exoskeleton compared with a human. *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 6728-6731). IEEE.

Conor James Walsh, K. E. (2007, March 8). A Quasi-Passive Leg Exoskeleton For Load-Carrying Augmentation. *International Jounral of Humanoid Robotics*, pp. 487-506.

Dasari Karthik, C. K. (2022). Identifying the significant factors affecting the masonry labour productivity. *Internal Journal of Construction Management*, 464-472.

Durda, F. (2021). *Serial and UART Tutorial.* FreeBSD Foundation.

ExRx. (n.d.). *Body Segment Data*. Retrieved from ExRx.net: https://exrx.net/Kinesiology/Segments

H. Kazerooni, J.-L. R. (2006). On the Control of the Berkeley Lower Extremity Exoskeleton (BLEEX). *Proceedings of the 2005 IEEE International Conference on Robotics and Automation.* Berlin: IEEE.

Hao Lee, P. W. (2019). Lower Limb Exoskeleton Systems—Overview. *Wearable Robotics*.

Kinoshita, H. (1985). Effects of different loads and carrying systems on selected biomechanical parameters describing walking gait. *Ergonomics*, 1347-1362.

M. Fontana, R. V. (2014). The body extender: a full-body exoskeleton for the transport and handling of heavy loads. *Robotics and Automation*, pp. 34-44.

M. Vukobratovic, B. B. (2006). Towards a unified understanding of basic notions and terms in humanoid. *Robotica*, 87-101.

Makinson, G. E. (1971). *Research and Development Prototype for Machine Augmentation of Human Strength and Endurance: Hardiman I Project.* The Company.

Martin Grimmer, A. A. (2020). Human Lower Limb Joint Biomechanics in Daily Life Activities: A Literature Based Requirement Analysis for Anthropomorphic Robot Design. *Humanoid Robotics*.

Michael R Tucker, J. O. (2015). Control strategies for active lower extremity prosthetics and orthotics: a review. *Journal of NeuroEngineering and Rehabilitation*.

Neumann, D. (2009). *Kinesiology of the Musculoskeletal System: Foundations for Physical Rehabilition.* Mosby.

Norazam Aliman, R. R. (2017). Design and development of lower limb exoskeletons: A survey. *Robotics and Autonomous Systems*, 102-116.

Robert V. Schulte, E. C. (2023). Database of lower limb kinematics and electromyography during gait-related activities in able-bodied subjects. *Scientific Data*.

Sai K. Banala, S. K.-L. (2006). Gravity-Balancing Leg Orthosis and Its Performance Evaluation. *IEEE Transactions on Robotics* (pp. 1228 - 1239). IEEE.

Sul, S.-J. K.-K. (1995, November 6). Direct Torque Control of Brushless DC Motor with Nonideal Trapezoidal Back EMF. *IEEE Transactions on Power Electronics*.

Vikash Kumar, Y. V. (2019). Review of Exoskeleton : History, Design and Control. *3rd International Conference on Recent Developments in Control, Automation & Power Engineering (RDCAPE)* (pp. 677-682). IEEE.

Walter Pirker, R. K. (2016). Gait disorders in adults and the elderly. *Wien Klin Wochenschr*.

Yagn, N. (1890). *US Patent No. 420179.*

Zhiguo Feng, J. Q. (2007). Biomechanical design of the powered gait orthosis. *IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 1698-1702). Sanya: IEEE.

APPENDIX A

ESCON Module 50/8 Capabilities

The controller and motherboard can control BLDC motors up to a power rating of 260W and can handle a maximum continuous current of 15A. It can take in the hall sensor and encoder inputs (double-ended, up to two channels) from the motor to compute rotor speed.

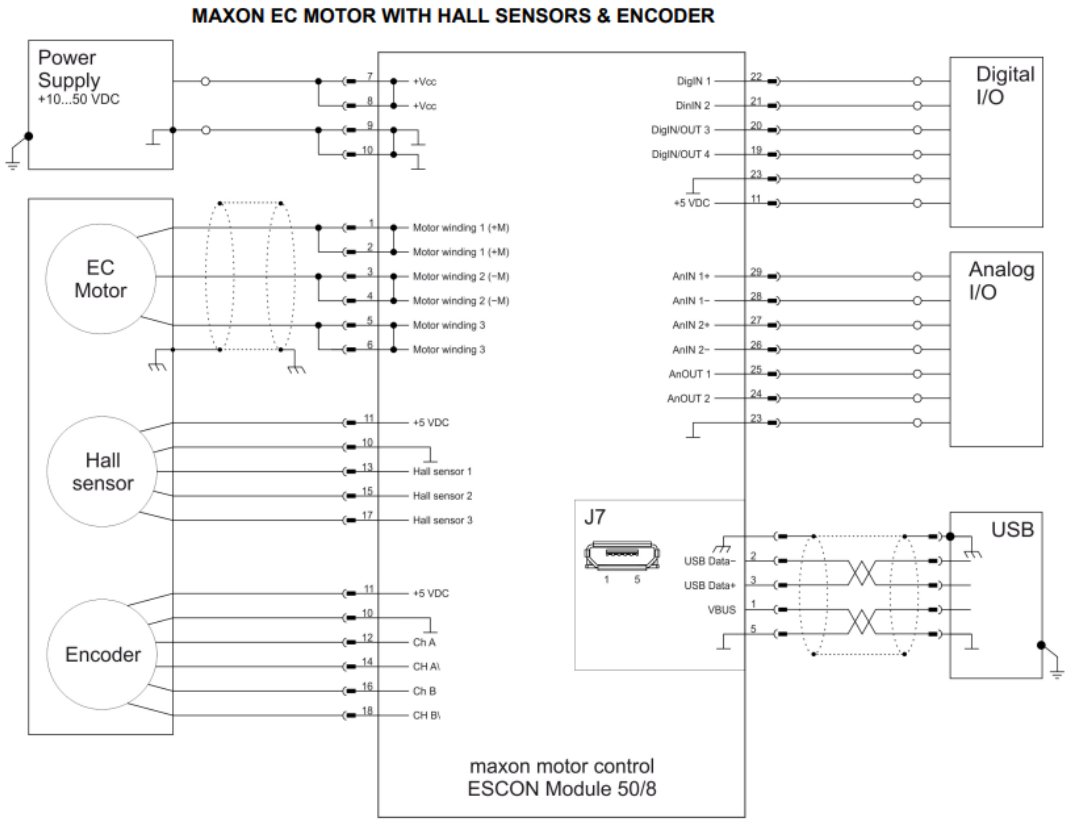


Figure A.1: Maxon controller schematic.

It has the following I/O capabilities:

i. Digital comparator outputs for current and speed.
ii. Digital input duty cycle measuring capability.
iii. Digital HIGH and LOW detection capability.

iv. Analog output of real-time actual speed, control signal speed, demand current, actual averaged current, and temperature.
v. Single and double-ended analog signal input detection and scaling.

Control Schemes:

i. Current Controller
ii. Speed Controller (Closed Loop)
iii. Speed Controller (Open Loop)
    a. Based on static IxR gain (reconfigurable)
    b. Based on adaptive IxR gain

Debug and tuning capabilities:

i. A control scheme can be chosen, and the inputs specified. Once done, the controllers can be auto-tuned by an internal system identification routine, or manual parameter input-based expert tuning can be done.
ii. Expert tuning can be done on the controller to adjust the controller's response from a slow response to a stiff response. A slider is given to choose the required damping of the control scheme.
iii. An internal system diagnostics toolbox that checks individual motor subsystems is present and can be used over USB to detect failures, possible causes, and their fixes.

APPENDIX B

BNO-055

The IMU used is BNO-055, a 9-DOF IMU made by Bosch Sensortec. It has low gyro drift with time and has reliable readings. It has an ARM Cortex-M0-based processor that performs computations on the sensor readings, abstracts the sensor-fused data into a class-based structure, and transmits the same via I2C/UART to a host MCU/MPU. It outputs the following readings:

i. Absolute Orientation (Euler Angle Vector based on 360° sphere)
ii. Absolute Orientation (Quaternion)
iii. Angular Velocity Vector (rad/s)
iv. Acceleration Vector (in m/s^2)
v. Linear Acceleration Vector (in m/s^2)
vi. Gravity Vector (in m/s^2)
vii. Magnetic Field Strength Vector (µT)
viii. Temperature (in °C)

APPENDIX C

Printed Circuit Board for the mid-level controller deployment

A PCB with a small form factor to be fitted with a mount on the links has been designed around the ESP32-WROOM-32-UE module, chosen based on the requirements of communication protocols, I/O pins, and compute ability to act as a mid-level controller. The pins chosen were such that they had PWM and digital I/O capabilities with no bootup glitches. Strapping and input-only pins were avoided. A USB-UART bridge was added alongside debug status LEDs to allow for programming the board. The provision for mounting a 48V to 5V buck converter to minimize the power sources used has been given and routed to ensure good current carrying capacity. The following considerations were taken during the design of the PCB:

- Delay matching on high-speed USB lines.
- Separating the clock and data lines of the I2C channel to avoid crosstalk.
- Copper pours, and thick traces for high current and voltage lines that can handle more than 1A were included.
- Inclusion of screw holes, pluggable JST connectors, and screw terminal positioning for ease of deployment on the user end.
- A MOSFET-based input power switching circuit has been provided to protect parent device communication ports when programming the module.

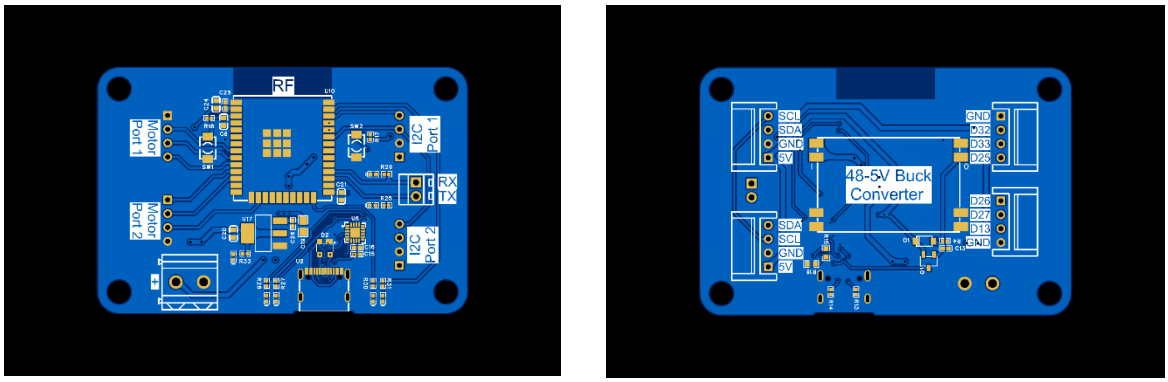


Figure C.1: PCB Footprint

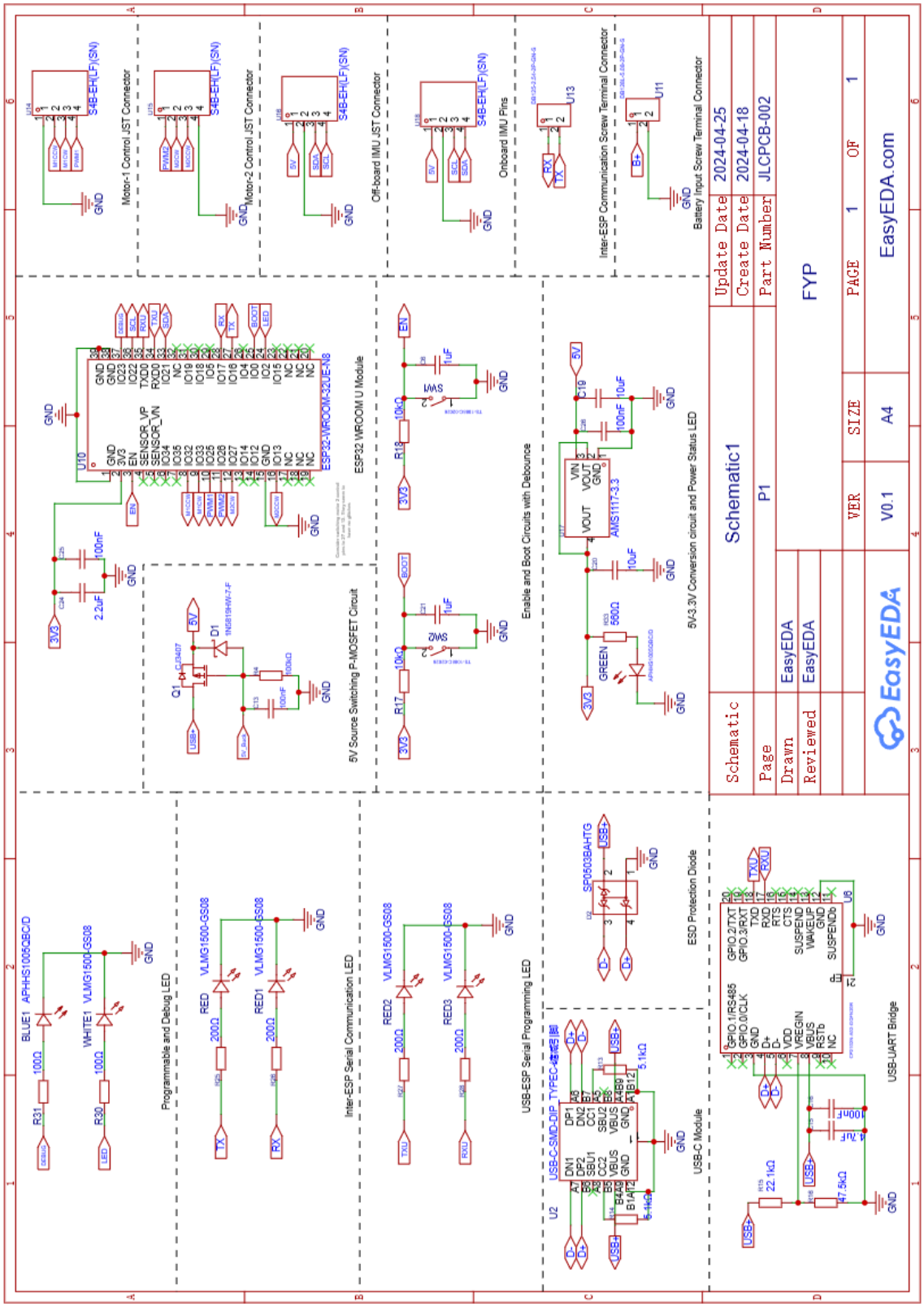


Figure C.2: PCB Schematic

APPENDIX D

Code for position control of single degree of freedom joint (shank)

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>

#define BOOT 0
#define VOLTAGE 25
#define CW 19
#define CCW 5

Adafruit_BNO055 bno = Adafruit_BNO055(55, 0x28, &Wire);

double controller_output, current_angle, prev_error, curr_error,
error_diff, set_point = 45;
double Kp = 2;
double Kd = 0;
int state = 0;

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);

  pinMode(CW, OUTPUT);
  pinMode(CCW, OUTPUT);
  pinMode(BOOT, INPUT);
    if (!bno.begin())
  {
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C
ADDR!");
    while (1);
  }

  delay(1000);
}

void loop() {
  sensors_event_t theta;
  bno.getEvent(&theta, Adafruit_BNO055::VECTOR_GRAVITY);
  get_angle(&theta);

  if (digitalRead(BOOT) == LOW) {
    if (state < 2) {
      state = state + 1;
    }
    else {
      state = 0;
```

```
    }
  }

  switch (state) {
    case 0:
      set_point = 45;
      break;
    case 1:
      set_point = 135;
      break;
    case 2:
      set_point = 60;
      break;
  }
  controller();

  int control_voltage = int(map(controller_output, 0, 720, 0, 255));
//  Serial.println(control_voltage);
  dacWrite(VOLTAGE, control_voltage);

  prev_error = curr_error;
  delay(10);
}

void controller() {
  curr_error = set_point - current_angle;
  error_diff = prev_error - curr_error;

  if (curr_error >= 0) {
    digitalWrite(CW, HIGH);
    digitalWrite(CCW, LOW);
  }
  else {
    digitalWrite(CW, LOW);
    digitalWrite(CCW, HIGH);
  }

  controller_output = abs((Kp*curr_error) + (Kd*error_diff));

}

void get_angle(sensors_event_t* event) {
  double x = -1000000, y = -1000000 , z = -1000000; //dumb values, easy
to spot problem
  event->type == SENSOR_TYPE_GRAVITY;
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
    if (abs(z) > 3) {
```

```
      Serial.println("Please hold the mount vertical.");
    }
    else {
      if ((x < 0) && (y < 0)) {
        current_angle = (atan(x/y) * 57.2957795131);
      }
      else if ((x < 0) && (y > 0)) {
        current_angle = 180 - (atan(-x/y) * 57.2957795131);
      }
      else if ((x > 0) && (y > 0)) {
        current_angle = 180 + (atan(x/y) * 57.2957795131);
      }
      else {
        current_angle = 360 - (atan(-x/y) * 57.2957795131);
      }
   }
}
```

APPENDIX E

Code for performing continuous motion on a two degree of freedom system (one full leg)

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>

#define SHIN_V 25
#define SHIN_CW 19
#define SHIN_CCW 23

#define THIGH_V 26
#define THIGH_CW 18
#define THIGH_CCW 4

const unsigned long stabilizing_time = 10000;
//const unsigned long thigh_time = 10000;
//const unsigned long shin_time = 9000;
const int freq = 5000;
const int shinChannel = 0;
const int thighChannel = 1;
const int debugChannel = 2;
const int resolution = 10;

unsigned long previous_time = 0;

double thigh_points[2] = {91, 105};
double shin_points[2] = {79, 75};
int i = 0;

Adafruit_BNO055 bno_thigh = Adafruit_BNO055(55, 0x28, &Wire);
Adafruit_BNO055 bno_shin = Adafruit_BNO055(55, 0x29, &Wire);

double Kp_thigh = 2, thigh_controller_output, thigh_angle, thigh_error;
double Kp_shin = 2, shin_controller_output, shin_angle, shin_error;

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);

  Serial.println("Setup initialized.");

  pinMode(SHIN_CW, OUTPUT);
  pinMode(SHIN_CCW, OUTPUT);

  pinMode(THIGH_CW, OUTPUT);
```

```
  pinMode(THIGH_CCW, OUTPUT);

  ledcSetup(thighChannel, freq, resolution);
  ledcSetup(shinChannel, freq, resolution);

  ledcAttachPin(26, thighChannel);
  ledcAttachPin(25, shinChannel);

  delay(10);

  Serial.println("Configured output and PWM pins.");

  if (!bno_thigh.begin()) {
    Serial.print("Problem with thigh BNO, rectify.");
    while (1);
  }

  if (!bno_shin.begin()) {
    Serial.print("Problem with shin BNO, rectify.");
    while (1);
  }

  delay(10);

  int setup_gap = 3;
  while (setup_gap > 0) {
    Serial.print("Setup complete. Executing loop in ");
    Serial.print(setup_gap);
    Serial.println(" seconds");
    delay(1000);
    setup_gap = setup_gap - 1;
  }
}

void loop() {

  controller();

  delay(50);
}

void update_values() {

  unsigned long current_time = millis();

  if (current_time - previous_time >= stabilizing_time) {
    i = 1 - i;
    previous_time = current_time;
  }
```

```
}


void feedback() {
  update_values();

  sensors_event_t theta_thigh, theta_shin;

  bno_thigh.getEvent(&theta_thigh, Adafruit_BNO055::VECTOR_GRAVITY);
  bno_shin.getEvent(&theta_shin, Adafruit_BNO055::VECTOR_GRAVITY);

  thigh_error = thigh_points[i] - get_angle(&theta_thigh);
  shin_error = shin_points[i] - get_angle(&theta_shin);
}


double get_angle(sensors_event_t* event) {
  double temp_angle, x, y, z;
  event->type == SENSOR_TYPE_GRAVITY;
  x = event->acceleration.x;
  y = event->acceleration.y;
  z = event->acceleration.z;

  if (abs(z) > 3) {
    Serial.println("Please hold the mount vertical.");
  }
  else {

    if ((x < 0) && (y < 0)) {
      temp_angle = (atan(y / x) * 57.2957795131);
    }
    else if ((x > 0) && (y < 0)) {
      temp_angle = 180 - (atan(y / x) * 57.2957795131);
    }
    else if ((x > 0) && (y > 0)) {
      temp_angle = (atan(y / x) * 57.2957795131);
    }
    else {
      temp_angle = 180 - (atan(-y / x) * 57.2957795131);
    }
  }
  if (temp_angle > 180) {
    temp_angle = temp_angle - 180;
  }
  return temp_angle;
}

void controller() {
```

```
  feedback();

  if (shin_error >= 0) {
    digitalWrite(SHIN_CW, LOW);
    digitalWrite(SHIN_CCW, HIGH);
  }
  else {
    digitalWrite(SHIN_CW, HIGH);
    digitalWrite(SHIN_CCW, LOW);
  }
  if (thigh_error >= 0) {
    digitalWrite(THIGH_CW, LOW);
    digitalWrite(THIGH_CCW, HIGH);
  }
  else {
    digitalWrite(THIGH_CW, HIGH);
    digitalWrite(THIGH_CCW, LOW);
  }

  shin_controller_output = abs(Kp_shin * shin_error);
  thigh_controller_output = abs(Kp_thigh * thigh_error);

  actuation();
}

void actuation() {
  int shin_duty, thigh_duty;
  if (abs(shin_error) > 2) {
    shin_duty = int(map(shin_controller_output, 0, 240, 102, 922));
  }
  else if (abs(shin_error) <= 2) {
    shin_duty = 102;
  }

  if (abs(thigh_error) > 2) {
    thigh_duty = int(map(thigh_controller_output, 0, 240, 102, 922));
  }
  else if (abs(thigh_error) <= 2) {
    thigh_duty = 102;
  }

  ledcWrite(shinChannel, shin_duty);
  ledcWrite(thighChannel, thigh_duty);
}
```

APPENDIX F

Python Script for extraction and plotting joint angles from the dataset

```
import os
import h5py
import matplotlib.pyplot as plt
import numpy as np
import csv

# Path to data files
path = '../Gait Data'
subj = 'MP104.hdf5'

# Load file
hf = h5py.File(os.path.join(path,subj))

# Load data from one trial and plot labels
trial = hf['Day_1']['Trial_01']

start = 30620
end = 31620

time = trial['Time'][start:end] - trial['Time'][start]

hip_l = []
hip_r = []
knee_l = []
knee_r = []
x_axis = []

for i in range(0, 1000, 50):
    hip_l.append(trial['Ang_Left_Hip'][start:end, -1][i])
    hip_r.append(trial['Ang_Right_Hip'][start:end, -1][i])
    knee_l.append(trial['Ang_Left_Knee'][start:end, -1][i])
    knee_r.append(trial['Ang_Right_Knee'][start:end, -1][i])
    x_axis.append(time[i])

hip_l.append(trial['Ang_Left_Hip'][start:end, -1][-1])
hip_r.append(trial['Ang_Right_Hip'][start:end, -1][-1])
knee_l.append(trial['Ang_Left_Knee'][start:end, -1][-1])
knee_r.append(trial['Ang_Right_Knee'][start:end, -1][-1])
x_axis.append(time[-1])

super_set = [hip_l, hip_r, knee_l, knee_r]

for set in super_set:
    for i in range(len(x_axis)):
```

```
        set[i] = round(set[i], 1)
for i in range(len(x_axis)):
    x_axis[i] = round(x_axis[i], 2)


csv_file = "new_angle_data.csv"
data = zip(x_axis, hip_l, hip_r, knee_l, knee_r)

with open(csv_file, 'w', newline='') as file:
    writer = csv.writer(file)
    # Write header
    writer.writerow(["Time", "Left Hip Angle", "Right Hip Angle", "Left
Knee Angle", "Right Knee Angle"])
    # Write data rows
    writer.writerows(data)

fig,ax = plt.subplots()
plt.title("Angles vs Time")
plt.xlabel("Time (seconds)")
plt.ylabel("Angle (degrees)")
plt.plot(x_axis, hip_r,'r')
plt.plot(x_axis, hip_l,'b')
plt.plot(x_axis, knee_r,'g')
plt.plot(x_axis, knee_l,'y')

plt.legend(["Right Hip", "Left Hip", "Right Knee", "Left Knee"])
plt.show()
```