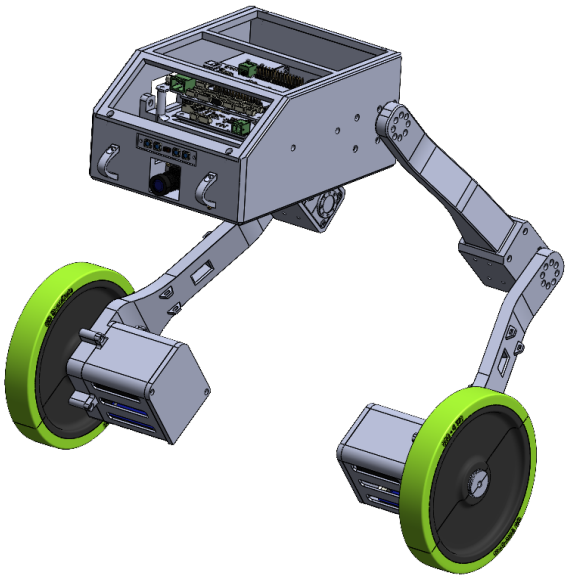
Institut für
Mechatronische Systeme
imes Leibniz Universität Hannover

Leibniz
Universität
Hannover

Design and Control of a Multi-Joint Two-Wheeled Inverted Pendulum Robot for Agile Motion Learning



Master's thesis 15-01/2024-1234

Mohamed Barakat
Matrikelnummer 10044038

Hannover, 15. January 2024

First Examiner	Prof. Dr.-Ing. Thomas Seel
Second Examiner	Prof. Dr.-Ing. Firstname Lastname
Supervisor	Dustin Lehmann

Institut für Mechatronische Systeme
Leibniz Universität Hannover

Leibniz Universität Hannover

15-01/2024-1234

Declaration of Authorship

Name:	**Mohamed Barakat**
Registration no.:	10044038
Thesis title:	Design and Control of a Multi-Joint Two-Wheeled Inverted Pendulum Robot for Agile Motion Learning
Type of thesis:	Master's thesis
Study program:	International Mechatronics
Submission date:	15. January 2024
First examiner :	Prof. Dr.-Ing. Thomas Seel
Second examiner:	Prof. Dr.-Ing. Firstname Lastname
Supervisor:	Dustin Lehmann

I, *Mohamed Barakat*, hereby affirm that the Master's thesis entitled *Design and Control of a Multi-Joint Two-Wheeled Inverted Pendulum Robot for Agile Motion Learning* was written independently, that no references and aids other than those indicated were used, that all passages of the thesis which were taken over literally or analogously from other sources are marked as such and that the thesis has not yet been presented to any examination board in the same or similar form.

I hereby agree to the transmission of my work also to external services for plagiarism checking by plagiarism software.

Hannover, 15. January 2024

(Mohamed Barakat)

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Ort, Datum	Mohamed Barakat

Abstract

The robotic industry is growing rapidly. Robots are becoming more and more capable and are used in a wide range of applications. The design, modeling and control of robotic systems is a research field that is constantly evolving.

However, there are still many challenges to overcome. The design of robotic systems is still a complex and time-consuming process, particularly in regard to multi-legged robots that should be able to execute complex maneuvers. The control of robotic systems is still a challenging task, especially when it comes to dynamic and complex systems. The search for the optimal control strategy is still an open research question.

This thesis presents an in-depth exploration of the dynamic control and design of robotic systems. The main focus is on the design and control of a multi-legged robot. The robot is designed and built from scratch, including the mechanical and electronic design. A combination of theoretical modeling and practical implementation were used to achieve the desired objectives. The research first focuses on the design of the robot, including the mechanical and electronic design. The robot is then modeled and simulated in a virtual environment. The simulation is used to test and evaluate the robot's performance. The robot is then built and tested in real life. The research investigates the main elements that affect maneuverability and stability, such as the integration of multiple degrees of freedom, structural integrity, and weight distribution.

Significant results are achieved in the design and control of the robot. The robot demonstrates the capability to maintain balance while adjusting its hip and knee joint angles, as well as exhibits stable locomotion on its two wheels, effectively navigating within the simulated environment. The robot's design is optimized to find the best trade-off between stability and maneuverability. Different problems are addressed throughout the research, such as the design complexity to meet the mobility requirements, Incorporating the changing location of the center of mass in the dynamic model, and retuning the control of the robot when one or both of the joint angles change. The developed Legged two-wheeled inverted pendulum robot serves as a platform for future research to build upon and test different control strategies.

This research contributes valuable insights into the field of robotics, offering practical solutions for the challenges faced in designing and controlling wheeled and bipedal robots. It lays a foundation for future innovations, aiming to broaden the scope and capabilities of robotic systems in real-world applications.

Keywords— Robotics, Multi-Legged Robot, Control, Simulation, Design, CAD, Modeling, Inverted Pendulum, Two-Wheeled bi-pedal legged robot

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Figure 1.1: Handle Robot by Boston Dynamics[1]

Robotics has always been a cutting-edge field that combines sophisticated engineering, artificial intelligence, and a comprehension of human-environment interactions. This study is motivated by multiple important elements, all of which highlight the importance and relevance of the research. The reasons for undertaking this research is the desire to advance the field of robotics, fill existing knowledge gaps, have a broad societal impact, and foster interdisciplinary collaboration. By developing a multi-legged robotic system with enhanced movement capabilities and control strategies, this research endeavors to set new standards in robotic design and functionality. Even with significant advancements, there are still many unanswered questions about robotic control and locomotion, particularly in systems that replicate biological structures and processes. Complex dynamic tasks that living beings handle with ease are sometimes difficult for traditional robotic systems to accomplish. This research aims to close these gaps by concentrating on a multi-legged robot that finds inspiration in the natural environment. This will provide insights into more realistic and effective movement tactics. The applications of such advanced robotic systems are vast and varied, ranging from search and rescue operations in hazardous environments to assistive technology in warehouse automation.By pushing the boundaries of what is currently possible in robotic design and control, this research has the potential to make significant contributions to fields where human intervention is limited, dangerous, or impractical. This project is inherently interdisciplinary, integrating concepts from mechanical engineering, computer science, control theory, and even biology. Such cross-disciplinary collaboration is crucial for driving innovation, as it allows for the exchange of ideas and methods from diverse fields. This approach is expected to yield novel solutions and advancements that could extend well beyond the scope of this project.

1.2 Explanation of the Goals and Requirements

In this research, we aim to develop a multi-legged robotic system that can perform complex movements and interact with its environment. The robot will be able to balance on two wheeled multi-jointed legs. The robot would be based on the previous TWIPR robot. Extra degrees of freedom will be added to the robot to allow for more complex movement. The robot would be designed from scratch to meet the requirements of the project. The robot would be designed using CAD software. The design would take into account the mechanical, electrical, and software requirements of the robot. Fabrication of the robot chassis would be done using 3D printing.Electrical design and assembly would be done using off-the-shelf components. Mathematical modeling and simulation of the robot would be done to determine the robot's dynamics and control. Different control strategies would be explored and implemented. Development of firmware and software for the robot would be done. The robot would be tested and evaluated in simulation and in real life. The robot would be tested for its ability to balance and move in different environments. The robot would be tested for its ability to perform complex movements and interact with its environment. ForThe exploration of experimental methods and different learning algorithms, we need cheep to build and easily be repaired robot.

2 Literature Review

2.1 Overview of Two-Wheeled Inverted Pendulum Robots (TWIPR)

Figure 2.1: Ascento Robot by ETH Zurich[2]

The Legged Two-Wheeled Inverted Pendulum Robot (LTWIPR) is a type of mobile robot that has two wheels and a body carried by two legs. The robot acts as an inverted pendulum, with the wheels acting as the pivot point. The robot is able to balance itself on the two wheels. The robot is able to move by tilting the body forward or backward, causing the wheels to rotate in the direction of the tilt. The robot is able to turn by moving the wheels in opposite directions. The robot is able to move in any direction by combining these movements. LTWIPRs are able to move in a variety of environments, including rough terrain and stairs. LTWIPRs are able to perform complex movements, such as jumping and climbing. LTWIPRs are able to interact with their environment, such as navigating around obstacles and diving under obstacles. LTWIPRs are able to perform tasks such as search and rescue, surveillance, and exploration.

2.2 Prior Works and Advances in Multi-Legged Robotic Systems

Multi-legged robots have been studied extensively in the past. Different types of multi-legged robots have been developed such as the quadruped robots, hexapod robots, and biped robots.

2.2.1 Mechanical Design and Development

Figure 2.2: Honda's Asimo, Boston Dynamics' Atlas and Sony QRIO[3][4][5]

It is obvious that the mass distribution of the robot is a key factor in determining the robot's stability. Various design considerations when designing a robot are important to consider, such as the robot's size, degree of freedom, link design, and mass distribution[8]. Biped robots have many parameters that affect their locomotion, making it difficult to determine the right design parameters to achieve stable and reliable gaits. The purpose of designing and assembling this biped robot is to advance research on humanoid robots and to offer a platform for studies on biped locomotion. It is simple to change the structure to suit the researcher's interests and the type of testing that must be done. As a result, the platform requires little upkeep. With the help of sensors, electrical actuators, and mechanical connections, biped robots with full functionality are put together to carry out a specific task. An apparatus for processing (PU), primarily consisting of an embedded processor, which methodically regulates every part of the biped robot. Several robots are built around these microcontrollers. Because of the significant processing power that is contained on a single chip, programmers have a great deal of flexibility[9]. The design of bipedal robots is a complex task that requires a thorough understanding of mechanics, control systems, human biomechanics, and robotics software.

Figure 2.3: evoBOT-3[6]

The Legged Two-Wheeled Inverted Pendulum Robot (LTWIPR) such as the Ascento robot by ETH Zurich [2] and similar robots designs studied in [10],[11],[12],[13] and [14] demonstrate the different design approaches that can be used to design a LTWIPR. The wheel-leg hybrid design of the LTWIPR allows for the robot to have the advantages of both wheeled and legged robots. another design differs, such as the evoBOT-3. High levels of flexibility and agility define the evoBOT. This collaborative robot can be used in complex urban spaces, going beyond the traditional logistical context. These extensions include load carriers and the passing and turning of objects.[6]

2.2.2 Dynamic Modeling of Wheeled Bipedal Robots

The dynamic modeling of wheeled bipedal robots is a complex task that requires a thorough understanding of the robot's kinematics and dynamics. [15] outline the kinematics and dynamics of the Two-wheeled Inverted Pendulum Balancing Mobile Robot where the kinematics constraints are derived and the dynamics are derived using the Lagrangian and Kane's methods. The equations of motions derived in this work can be considered as the base for the Legged Two-Wheeled Inverted Pendulum Robot (LTWIPR) since the Two-wheeled Inverted Pendulum Balancing Mobile Robot is a generalization of the LTWIPR. Free body diagrams are also different approaches that can be used to derive the equations of motion of the LTWIPR. The study [16] presents the modeling for the TWSB robot using free body diagram approach. The modeling of the robot is a critical step in the development of the robot since it is used to derive the equations of motion of the robot which are used in the control of the robot. Good understanding of the robot's dynamics is required to develop a good control strategy for the robot.

2.2.3 Control of Wheeled Bipedal Robots

Different control strategies have been implemented on wheeled bipedal robots such as Fuzzy Logic, PID, LQR, MPC, and Reinforcement Learning. The control strategies are chosen based on different factors such as the control objective, the complexity of the Robot's Dynamics and Requirements, The Robot's Operating Environment. In [13] it shows that To keep the WBR balanced while it is standing and moving on the ground, an intelligent motion and balance controller (IMBC) built on a fuzzy logic technique is suggested. It is important to note that the suggested IMBC system does not require any prior understanding of system dynamics, and that human knowledge's qualitative components are used to adjust the controller parameters. The study [12] shows the implementation of LQR control on a LTWIPR because it efficiently resolves the issue of striking a balance between control effort and good system response. The wheels of the Wheel-based robot are controlled by the LQR controller because the pendulum length varies during movement. Other studies demonstrate classical control strategies such as PID control on Two-wheeled self-balance robot. The study [16]shows the implementation of PID control on a Two-wheeled self-balance robot where PID gains are tuned until ideal performance is achieved. Some other methods such as controlled lagrangian are used to stabilize an inverted pendulum that is fastened to the end of a robotic arm that rotates[17]. Model Predictive Control (MPC) is also a powerful and versatile control strategy that has been used on wheeled self-balancing robots. [18] presents the development of model predictive control (MPC) for each subsystem of a two-wheeled self-balancing robot (TWSBR) in order to achieve a stable and robust control system. The control of the robot is an intricate process that involves advanced algorithms to achive the desired control objective.

2.2.4 Integration of Learning and Adaptive Control Mechanisms in Wheeled Bipedal Robots

In the absence of an accurate dynamic model of the robot, Adaptive dynamic programming (ADP) and reinforcement learning (RL) can be used to derive an adaptive optimal control solution based on learning. In contrast to the LQR control which requires an accurate dynamic model of the robot. In complex mechanical structures such as the LTWIPR, the dynamics are difficult to model accurately. Learning and adaptive control mechanisms can be used to overcome this challenge. The study [19]proposes a data-driven algorithm to generate near optimal balance control for a two-wheeled self-balancing robot. In other research, Adaptive variable impedance control (AVIC) is used to control the robot's balance specifically in the case of interaction with complex unknown terrain[20]. The AVIC controller is used to control the robot's balance by minimizing the force tracking error for each leg that is exerted on the body.

2.2.5 Applications and Future Directions

There are many applications for wheeled bipedal robots, including Agriculture, Search and Rescue, surveillance, warehouse automation, military, and others. Further research is needed to improve the performance of wheeled bipedal robots in terms of speed, stability, and robustness. Solutions for different problems such as the localization of the robot, the navigation of the robot, and the interaction of the robot with its environment are needed to advance the field [21]. Machine learning and artificial intelligence can be used to improve the performance of wheeled bipedal robots[22].

3 Mechanical Design

Modeling In this chapter, the details of the mechanical design are presented. where it goes from the initial conceptual design to the final design showing in the process the design decision-making for each critical point, This chapter emphasis the detailed description of the precise placement and alignment of different components such as the motors, wheels, battery, boards, and others to maintain the seamless integration of all the components into the robot body.

The mechanical design serves as an important pillar to define the new physical form, size, and shape of the TWIPR. Depending on how these criteria are defined, the robot would interact with the environment. taking into account that the design directly influences the center of gravity, which is crucial to consider in our robot due to the inverted pendulum nature to be able to balance and maintain the upright position. In addition to the impact that the design has on the maneuverability of the robot and how it would respond to the control signals to be able to execute a task.

3.1 Design Objectives and Requirements

The main objective is to come up with a new design for a multiple joints Robot to perform complex dynamic movements. This robot would be based on the Two wheeled inverted pendulum robots. The new design would add more degrees of freedom to enable the more complex movements. This enhances the robot's capabilities where it can execute more diverse scenarios. Initially, the main requirement was to add two more degrees of freedom where originally it used to have one degree of freedom in the wheels. The new design has three degrees of freedom, one in the wheels, one as a knee joint and one as a hip joint. The current design has two identical legs.

3.2 Conceptual Design

As for the initial design concepts the body was that main point of focus as shown in the figure 3.1.Three initial designs where considered mainly for the body. Symmetrical vertical body in figure A, symmetrical horizontal body in figure B and leaning forward body in figure C. For the three designs, two independent legs where considered. two designs where considered for the legs, the normal joint leg and the compliant leg

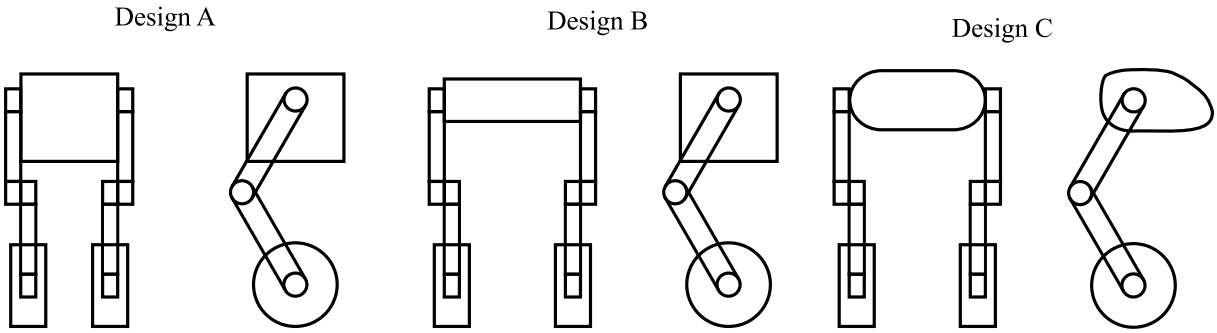


Figure 3.1: Three initial Design Concepts for the robot body

Two main designs where considered for the legs as shown in the figure 3.2, the normal joint leg and the compliant leg. The compliant leg is more flexible and can be used to absorb the shock from the ground. The normal joint leg is more rigid and can be used to generate more torque.in addition, the normal is more relative to our use-case as it can precisely control the position of the leg.

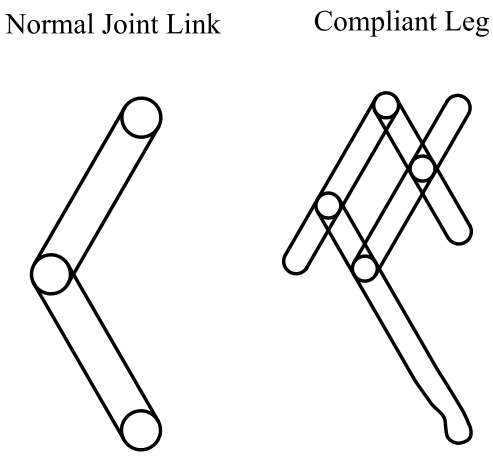


Figure 3.2: Leg Design Concepts

3.3 Schmatic Representation of the Robot

3.3.1 Schmatic overview of the robot

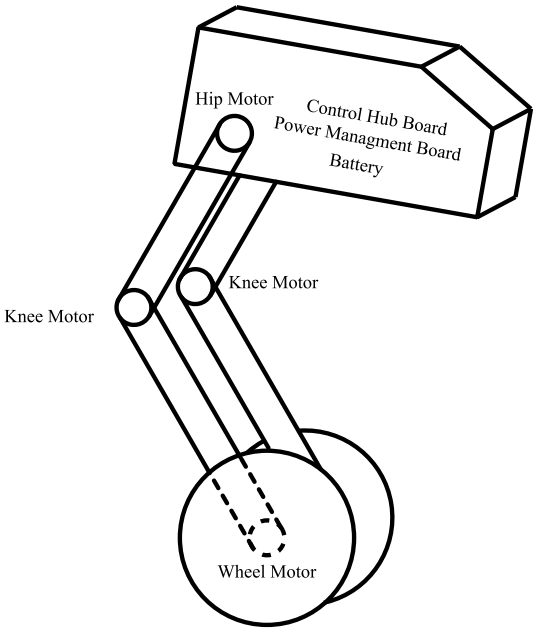


Figure 3.3: schmatic overview of the robot

As shown in the figure 3.3, In order to achieve the desired movements using the new articulated legs, then we need three motors on each leg, one motor for the wheel, one motor for the knee joint and one motor for the hip joint. In addition to the motors, the robot needs a battery to power the motors, a control board to control the motors, and a power management board to manage the power distribution between the different components. Some of the components are passed on from the previous TWIPR robot such as the control board, the power management board. For more information about the electronic components and there specifications, please refer to the electronics design chapter.

3.4 Initial Calculations

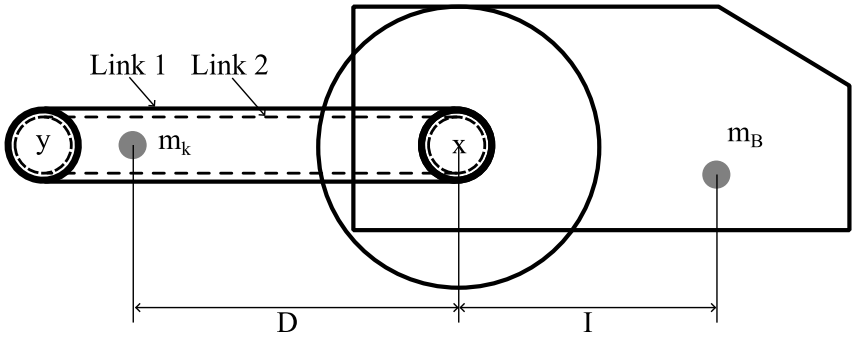


Figure 3.4: Initial Calculations

One of the main critical positions for the robot, as shown in the figure 3.4 is the position where the center of mass of the body and the center of mass of the legs are the furthest away from point x on the horizontal axis.It is important to make the initial torque calculations to be able to select the right motors for the robot. Starting from that position first, the robot should be able to balance itself and maintain

the upright position. secondly, the robot should be able to change the knee angle to lift the body while maintaining its balance. The calculations are based on some assumptions and simplifications, Considering only half the body weight and one leg. As shown in the figure, The Wheel motor shaft is aligned with the hip joint, the minimum torque required to balance the robot is calculated as follows:

The summation of the torques around point x should be equal to zero to maintain the balance with minimum motor torque.

$$\sum_{i=1}^{n} \tau_i = 0 \tag{3.1}$$

$$\sum_{i=1}^{n} \tau_i = m_B g I - m_K g (0.7) D \tag{3.2}$$

$$\sum_{i=1}^{n} \tau_i = 0.704(9.81)(0.03) - 0.2(9.81)(0.7)(0.15) \approx 0\text{Nm} \tag{3.3}$$

The Lengths of D and I can be modified to make sure that the summation of the torques around point x is equal or approximately equal to zero. This will make sure that the robot can balance itself with minimum wheel motor torque. The torques can cancel each other out by readjusting the lengths of D or I and also by modeling the weight distribution in the body and the legs.

The torque required to lift the body is calculated around the knee joint point y as follows:

$$\sum_{i=1}^{n} \tau_i = m_B g (D + I) - m_K g (0.3) D \tag{3.4}$$

$$\sum_{i=1}^{n} \tau_i = 0.704(9.81)(0.03 + 0.15) - 0.2(9.81)(0.3)(0.15) \approx 1.154\text{Nm} \tag{3.5}$$

1.154 Nm is the minimum torque required to hold the body in position. The Knee motor should be able to generate more torque to be able to lift the body and change the knee angle.

The approximate masses of the body and the legs are estimated in the table3.1. After manufacturing the parts and assembling the robot, the actual masses of the body and the legs are measured and the calculations are updated accordingly.

Table 3.1: Initial Calculations Assumptions

Parameter	Value	Description
m_B	0.704 kg	Mass of the body
m_K	0.2 kg	Mass of the Knee including the two legs
L_1	0.15 m	Length of Link 1
L_2	0.15 m	Length of Link 2
D	0.03 m	distance from the Wheel axis to the center of mass of leg
I	0.03 m	distance from the hip joint axis to the center of mass of the body

Note 3.1

To reduce the needed torque to lift we can:

- Shorten the links
- Reduce the body weight
- Use gearbox to increase the torque

3.5 Detailed Design Development

Throughout the design process, modularity and ease of assembly were considered. The design was broken down into three main parts: the body, the hip-knee link, and the knee-wheel link. Print iterations were made to ensure that the parts fit together and that the robot could be assembled with ease. Modifications were made to simplify the printing process and reduce the support material used.

3.5.1 Body Design

The body design is the main part of the robot, it is the main structure that holds all the components together. The components that are mounted on the body are the hip motors, the battery, camera, sensor, a rack that includes the power distribution board, the microcontroller board attached to the Raspberry Pi. The optimization of the body design is crucial to be able to fit all the components in a compact form and at the same time distribute the weight equally to maintain the balance of the robot. Different design features were considered, such as the cable management, the fastening features specifically designed to easily mount the battery, organizing the cable between the boards and the rest of the robot parts. Opening slots were to one side of the body to be able to access the boards and connect charging cables to the battery.

Figure 3.5: TOP view of the Body Design

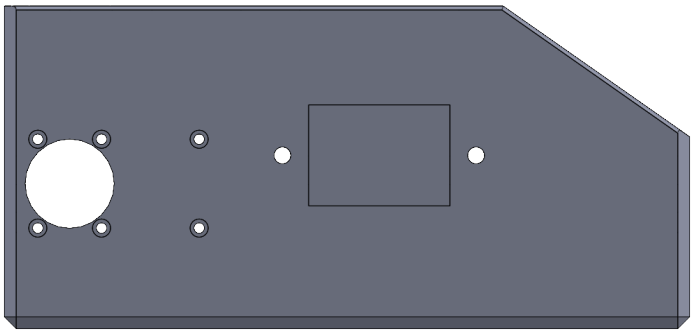


Figure 3.6: side view of the Body Design

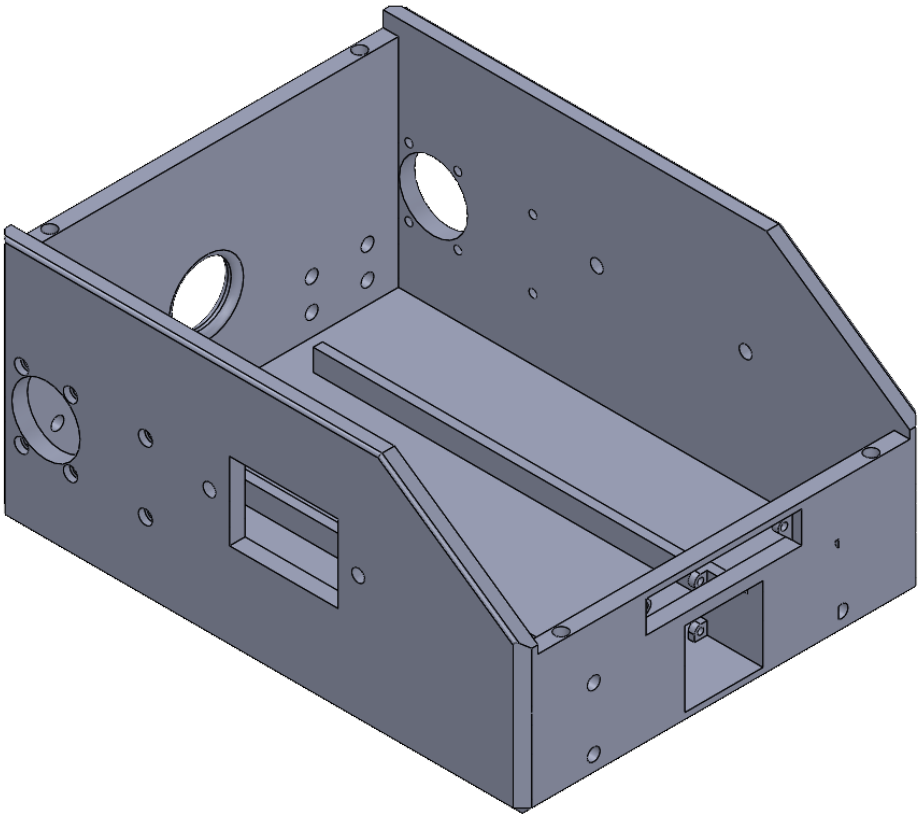


Figure 3.7: 3D view of the Body Design

3.5.2 Hip Knee Link Design

This link connects the hip body axis to the knee axis. The design of this link includes the knee motor housing form one-side and a frame to attach to the output horn of the hip motor. The motor housing is designed to ensure the fixed placement of the motor inside the link, As shown in the figure, 3.8 Cable management is also considered in the design of this link to make sure that the cables are not interfering with the movement of the link.The curvature of the link creates enough clearance so that when the hip axis is aligned with the wheel axis, they would not touch with each other even when considering the elastic bending of the legs.

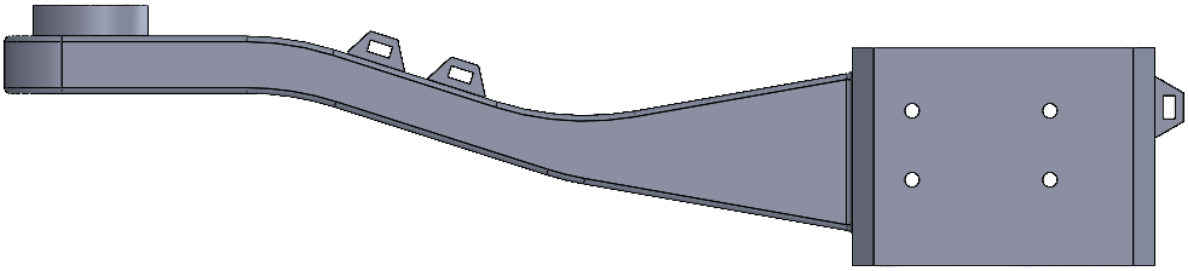


Figure 3.8: TOP view of the Body Knee Link

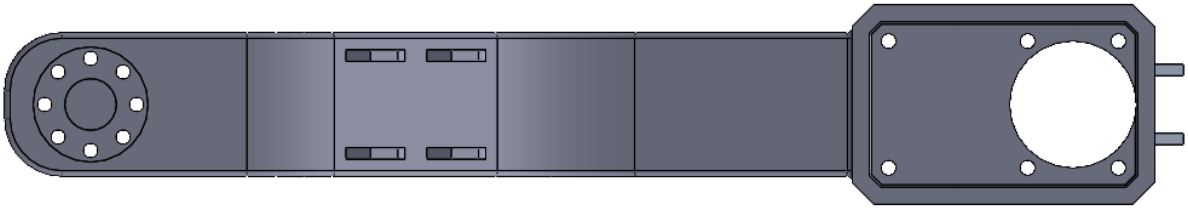


Figure 3.9: side view of the Body Knee Link

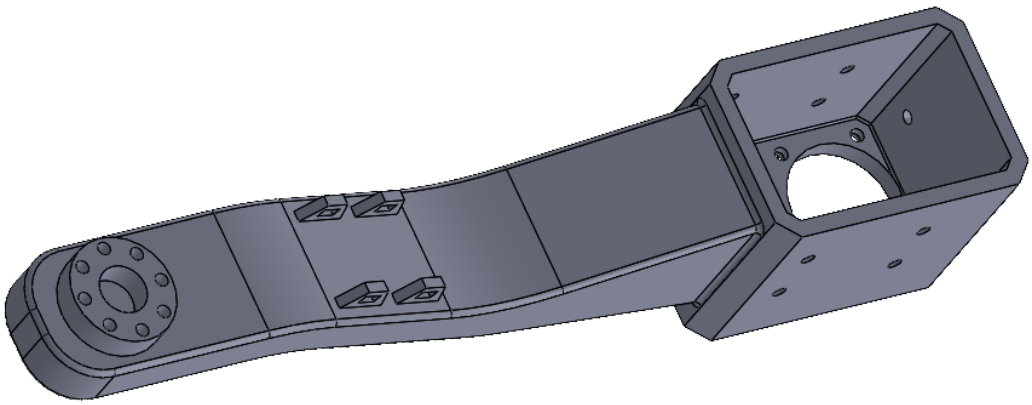


Figure 3.10: 3D view of the Body Knee Link

3.5.3 Knee Wheel Link Design

This link connects the knee axis to the wheel axis. The design of this link includes the wheel motor mounting form one-side and a frame to attach to the output horn of the knee motor from the other side. The motor mounting is designed to ensure the fixed placement of the motor on the link, As shown in the figure, 3.12. Cable management is also considered in the design of this link by adding a cable fastener and holes to pass the cables through the link.

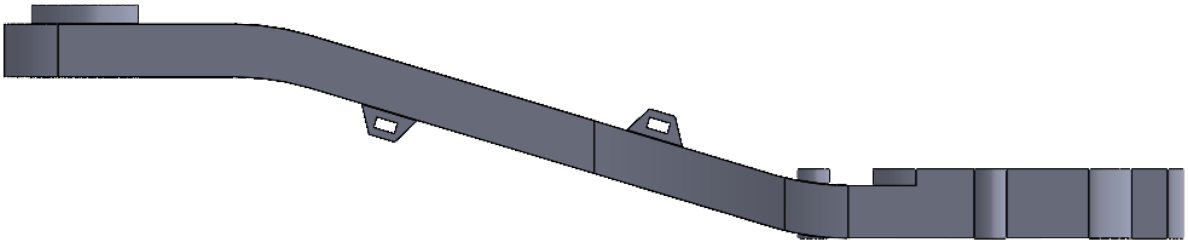


Figure 3.11: SIDE view of the Knee Wheel Link

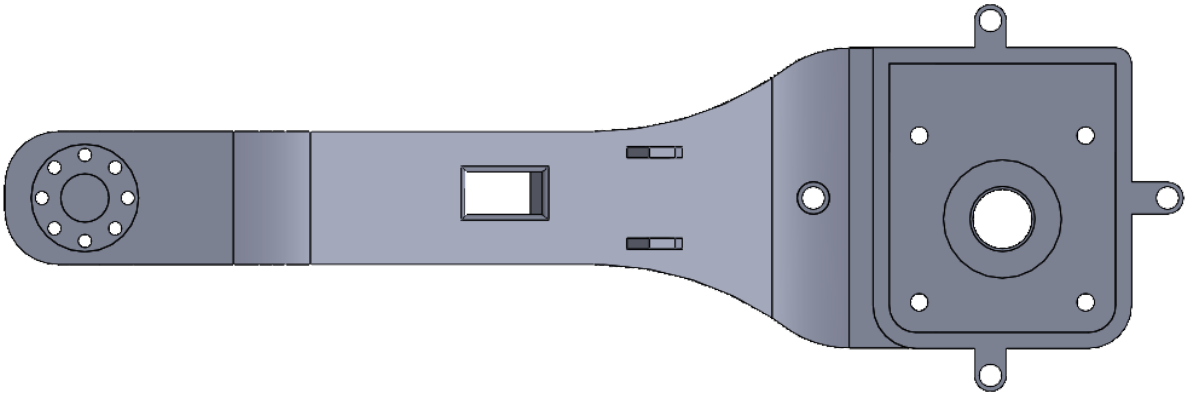


Figure 3.12: TOP view of the Knee Wheel Link

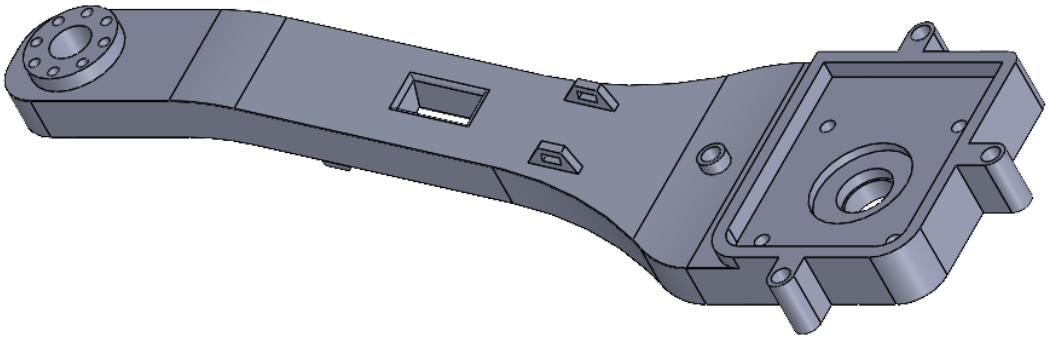


Figure 3.13: 3D view of the Knee Wheel Link

3.5.4 Boards Mounting rack

As shown in the figure 3.14,3.15,3.16, the board mounting rack is designed to hold the power distribution board, the microcontroller board, and the Raspberry Pi. The rack is designed to be mounted in the body of the robot by using four screws. Appropriate space is considered between the two boards as well as the clearance between the rack and the battery underneath it and the body cover on top of it.

Figure 3.14: Side view of the PCB Rack

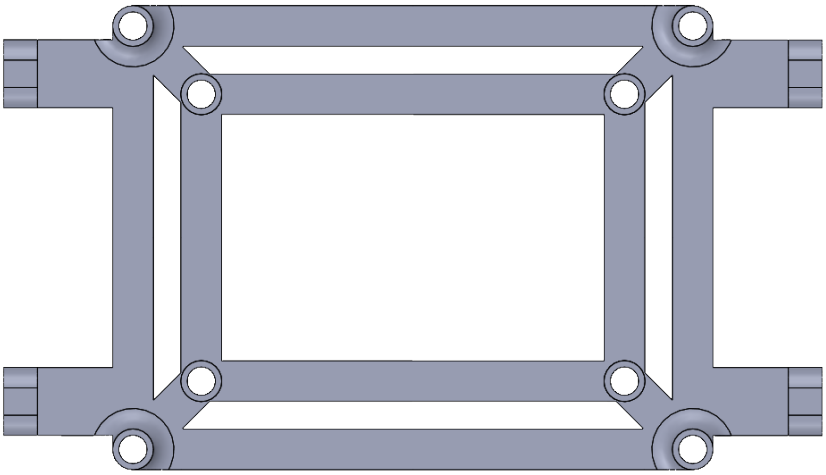


Figure 3.15: Top view of the PCB Rack

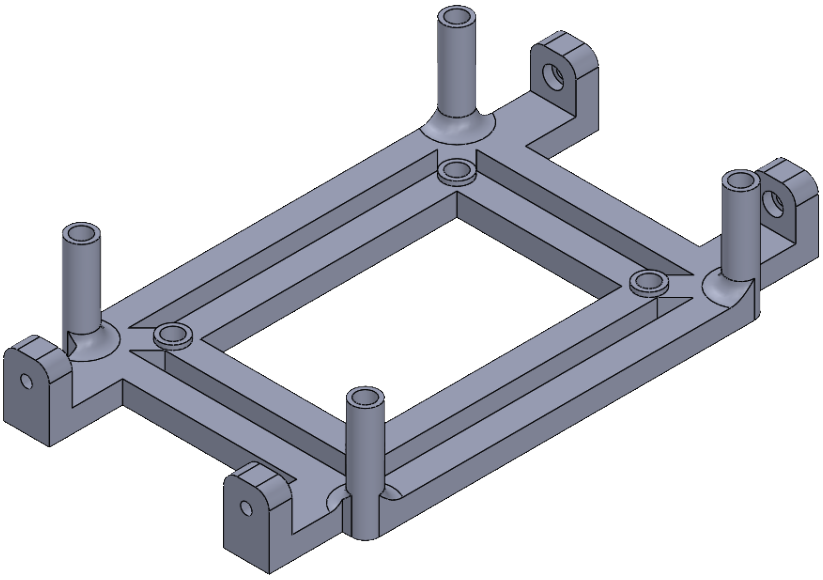


Figure 3.16: 3D view of the PCB Rack

3.5.5 Motor Cover

The motor cover is designed to protect the motor from any external objects that might interfere with the motor operation. The motor cover is designed to be mounted on the wheel knee link with the consideration of the clearance between the motor cover and hip knee link. Slots are designed to dissipate the heat generated from the motor as shown in the figure 3.17, 3.19.

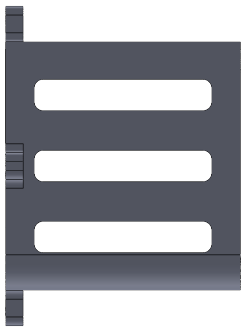


Figure 3.17: Side view of the Motor Cover

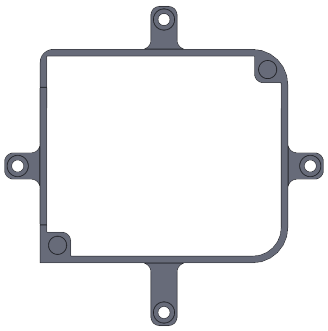


Figure 3.18: Top view of the Motor Cover

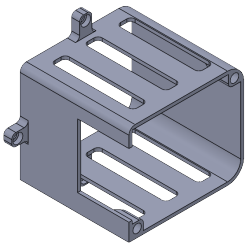


Figure 3.19: 3D view of the Motor Cover

3.5.6 Full Design

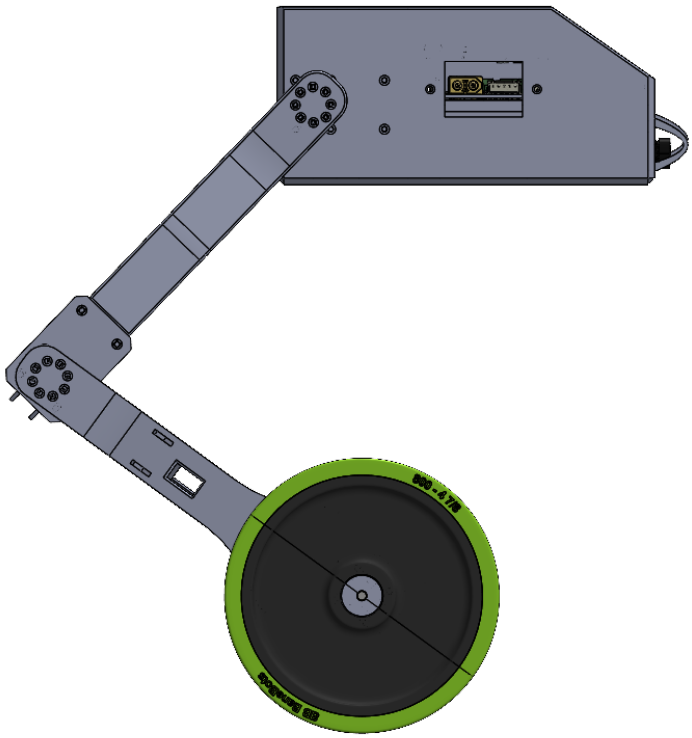


Figure 3.20: Side view of the Robot Assembly

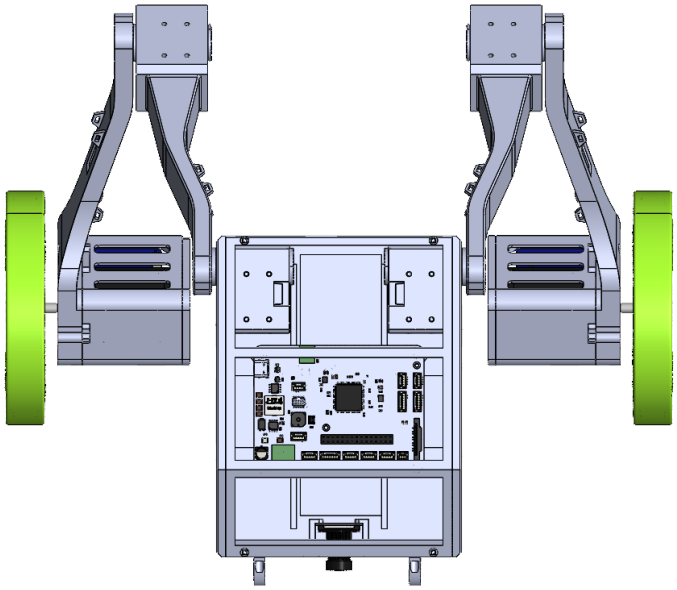


Figure 3.21: Top view of the Robot Assembly

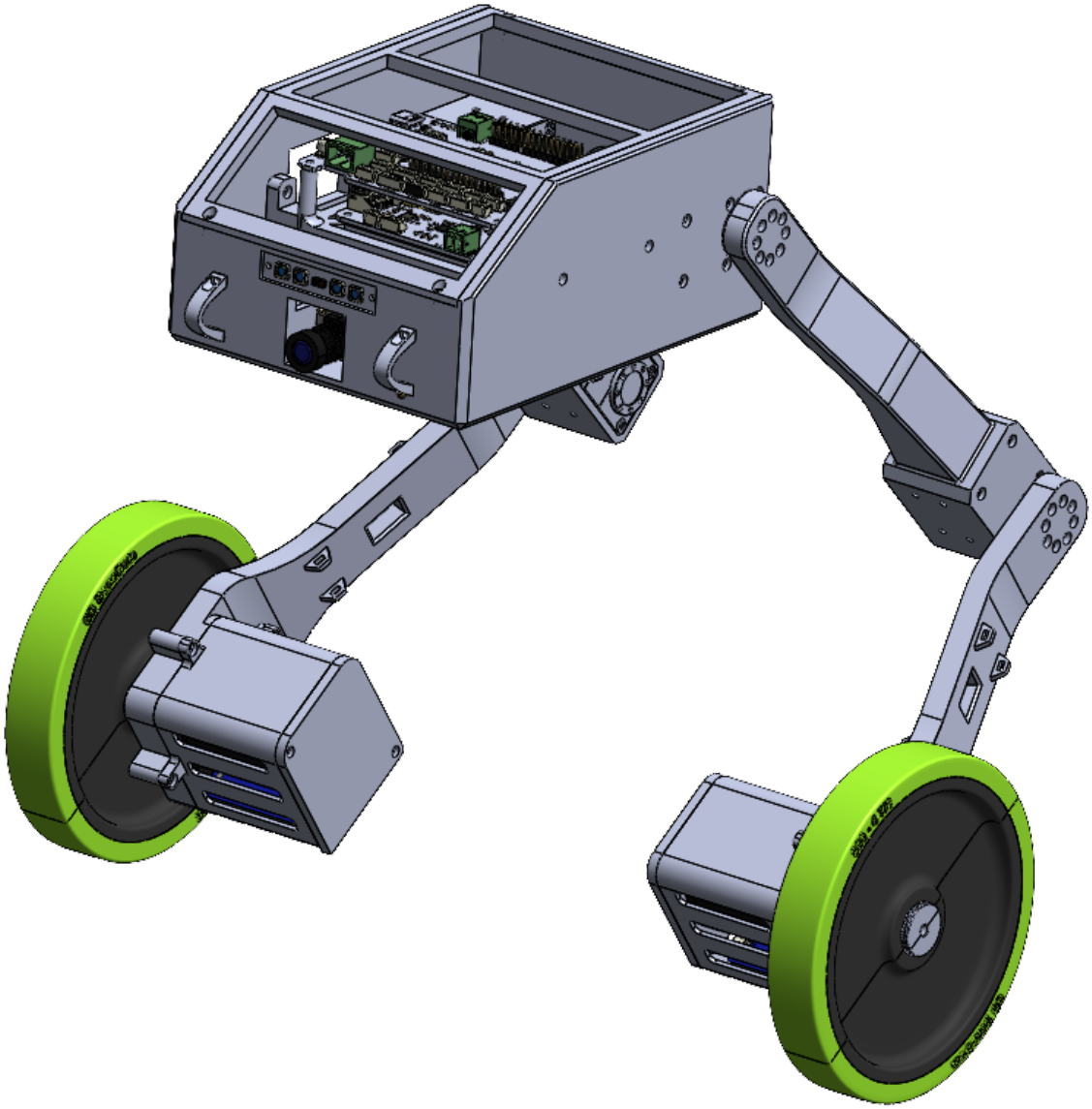


Figure 3.22: 3D view of the Robot Assembly

The full design of the robot is shown in the figure3.22. The robot is designed to be assembled in a modular way. The full assembly reveals how all the parts fit together. This includes the body, the hip-knees, the knee-wheel links, the motor cover, the boards mounting rack,the hip motors, the knee motors, the camera, the sensor, the face shield, the body cover,the BaneBots shaft hub, and the wheels. After assembling the robot, different movements configurations are tested using the CAD software to make sure that the robot can move freely without any interference between the different parts.

3.5.7 Dimensions

As shown in the figure 3.23, 3.24, the main dimensions of the robot are shown. Comparing the dimensions of the robot to the dimensions of the previous TWIPR robot, the new robot is much bigger in size because of the addition of the two legs. Compactness was not a main concern in the design of the robot, the main concern was to make sure that the robot can perform the required movements and interact with the environment.

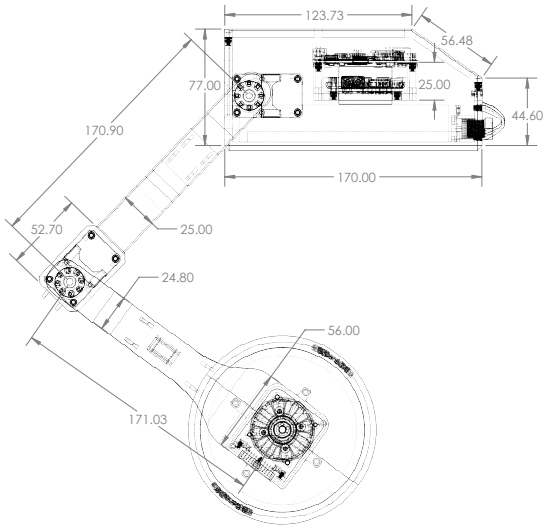


Figure 3.23: Side view of the Robot Assembly

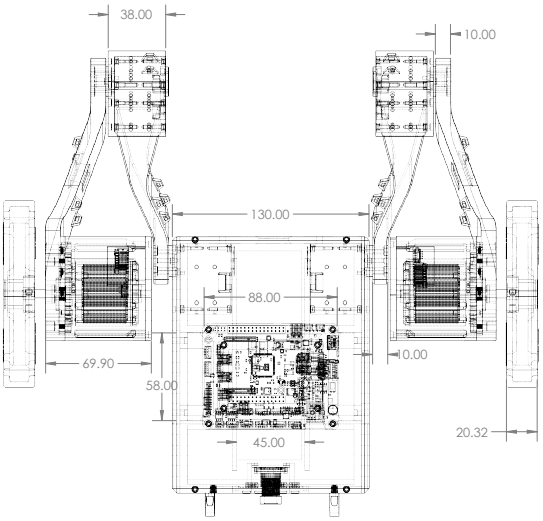


Figure 3.24: Top view of the Robot Assembly

3.5.8 Safety considerations

Safety is a crucial aspect to consider in the design of the robot. The robot is designed to be safe to operate in the environment and safe to interact with humans. Clearances are considered in the design to make sure that moving parts wouldn't interfere with each other. The motor cover is designed to protect the motor from any external objects that might interfere with the motor operation.

(a) Face Shield Side View

(b) Face Shield 3D View

Figure 3.25: Face Shield

The face shield of the body is designed to protect the components inside the body. The face shield is designed to be flexible to absorb the shock from any external objects that might hit the robot. It is screwed with two screws to the body. The cable management is considered in the design of the robot to make sure that the cables are routed from and to all the components in the robot in a safe way to protect the cables form tearing or getting damaged as well as to protect the cables from interfering with the movement of the robot.

3.6 Design for Manufacturability and Assembly

Design for manufacturability and assembly was considered throughout the design process. Design consideration was taken into account to make sure that the robot parts can be printed using 3D printing technology. Considering the limitations of 3D printing technology, the design was modified to make sure that the parts can be printed with acceptable quality. The orientation at which the parts are printed was considered while designing the parts. Assembly fitting and clearance were considered to make sure that the parts can be assembled with ease.

3.7 Prototyping and Iterative Design

During the design process, the sketches for the different parts were drawn relying on each other so that future modifications can be easily applied. The design was prototyped using 3D printing. Therefore, different problems were encountered during the prototyping process. The problems were analyzed and the design was modified accordingly to solve the problems. Some of the problems were related to the 3D printing process, and some were related to the design itself, such as the clearance between the parts, the cable management dimensions. The actual physical parts were slightly different in dimensions than the CAD model. This is why ititrations were made to the design.

4 Electronic Design

In this chapter, the details of the electronic design are presented. Where it emphasizes the details of the components' technical specifications and the selection process. The chapter also discusses the circuit design and the PCB design. In addition, the chapter discusses the power management and the power distribution.

The electronic design serves as a critical link between the robitic conceptual framework and the physical implementation of the robot.The electronic design translates the abstract control algorithm into tangible action and responses. The electronic design directly influences the performance, responsiveness, adaptability to various scenarios.

4.1 Design Objectives and Constraints

The main objectives of the electronic Design are to connect the different components and enable them to perform the desired tasks. The electronic design is responsible for controlling the motors. Robust communication between the electronic components is needed to ensure reliable operation of the robot. The electronic design is constrained by the power requirements of the motors. The motors require a high current to operate, and the electronic design should be able to provide the required current. In addition, the electronic design is constrained by the size of the components and their placement in the robot body.

4.2 Component Selection

The component diagram is shown in a figure 4.1shows the different components and their relationship with each other. The main components are the microcontroller, rassberry pi, the motors, Distance sensor, Camera, the battery. The motors are chosen based on the calculations done in the mechanical design chapter. The microcontroller is chosen based on the complixity of the control algorithm and the processing power needed to run the control algorithm.

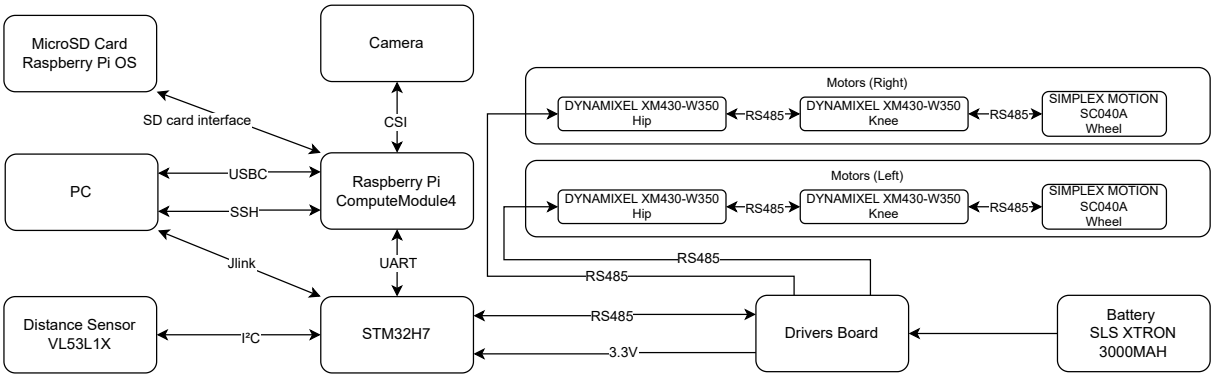


Figure 4.1: Components Diagram and there realationship with each other

4.2.1 Hip and Knee Motors

The DYNAMIXEL XM430-W350 is chosen as a motor for the hip joint and the knee joint. The motor is chosen because it has a high torque to weight ratio and it has a high resolution of 4096 steps per revolution. The motor has a built-in driver and it can be controlled using a serial communication protocol. The motor has a built-in encoder that can be used to measure the position of the motor. The motor has a maximum torque of 3.5 Nm and a maximum speed of 46 RPM. The motor has a maximum current of 2.1 A and a maximum voltage of 12 V. The motor has a weight of 82 g and a size of 28.5 x 46.5 x 34 mm.

Figure 4.2: DYNAMIXEL XM430-W350

4.2.2 Wheel Motors

The SIMPLEX MOTION SC040A is chosen as a motor for the wheels. The motor is chosen since it has a output of 120W and 280 mNm torque at 4000rpm. The motor has a built-in driver and it can be controlled using RS485 serial communication protocol. The motor has position and speed control with torque limit. The motor has a maximum torque of 800 mNm.

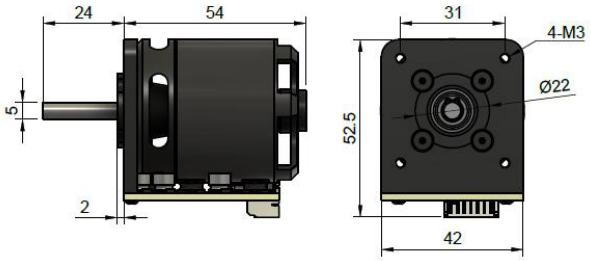


Figure 4.3: SIMPLEX MOTION SC040A

4.2.3 Raspberry Pi

The Raspberry Pi 4 compute module is chosen since it offers several advantages such as remote ssh connection via wifi. It can directly pass new control parameters to the microcontroller. It can be used to stream the video from the camera. The Raspberry Pi 4 compute module has a 64-bit quad-core ARM Cortex-A72 processor running at 1.5 GHz. It has 4 GB of LPDDR4-3200 SDRAM. The Raspberry Pi 4 compute module has a 32 GB eMMC Flash memory, a maximum current of 3 A and maximum voltage of 5.1 V.

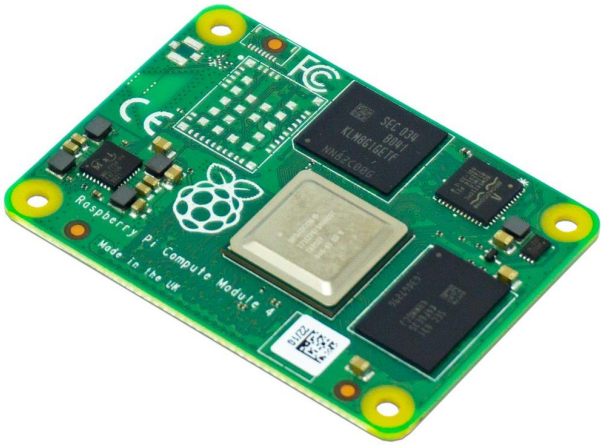


Figure 4.4: Raspberry Pi 4 compute module

4.2.4 Distance Sensor

The VL53L1X is chosen as a distance sensor for the robot. The sensor is chosen because it has a high accuracy and it has a high range. The sensor has a maximum range of 4 m. The sensor has a maximum accuracy of 1 mm. The sensor has a maximum field of view of 27 degrees. The sensor has a maximum current of 20 mA. The sensor has a maximum voltage of 3.6 V. The sensor has a weight of 1.6 g and a size of 4.4 x 2.4 x 1.0 mm.

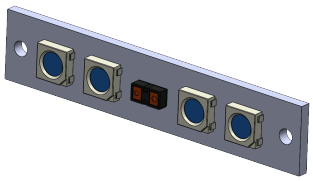


Figure 4.5: VL53L1X

4.2.5 Camera

The Raspberry Pi Camera Module V2 is chosen as a camera for the robot. The camera is chosen because it has a high resolution and it has a high frame rate. The camera has a resolution of 8 MP. The camera has a frame rate of 30 fps. The camera has a maximum current of 250 mA. The camera has a maximum voltage of 3.3 V. The camera has a weight of 3.4 g and a size of 25 x 23 x 9 mm.

Figure 4.6: Raspberry Pi Camera Module V2

4.2.6 Battery

The SLS XTRON 3000MAH 4S1P 14.8V 35C LIPO BATTERY is chosen as a battery for the robot. The battery is chosen because it has a high capacity and it has a high discharge rate. The battery has a capacity of 3000 mAh. The battery has a discharge rate of 35 C. The battery has a maximum current of 105 A. The battery has a maximum voltage of 16.8 V. The battery has a weight of 300 g and a size of 135 x 42 x 30 mm.

Figure 4.7: SLS XTRON 3000MAH 4S1P

4.3 Circuit Design

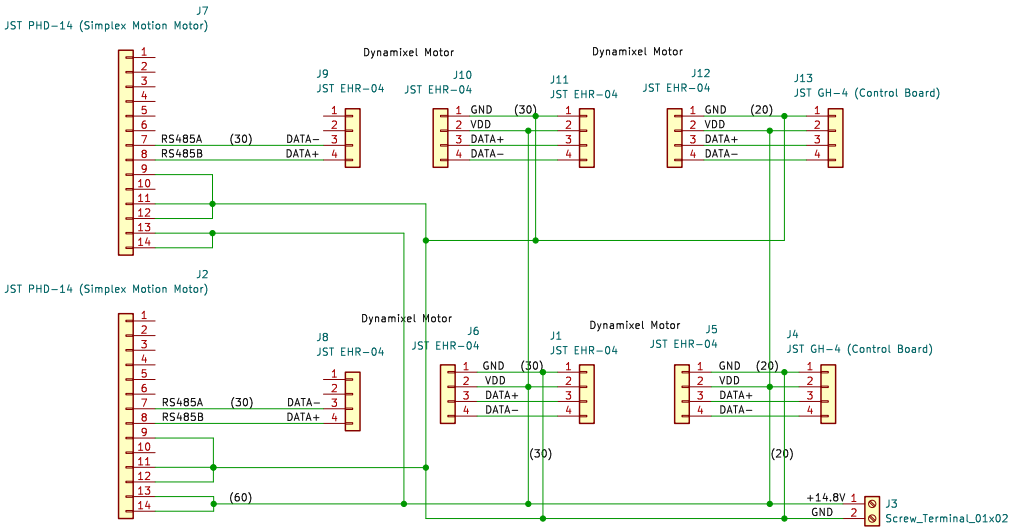


Figure 4.8: Electronic Design

The wiring tree of the electronic components is shown in figure 4.8 where it shows the connection between the different components and the microcontroller.The first motor of each leg is connected to the microcontroller and the rest of the motors are connected in series to the first motor of using daisy chain connection. the motors have the drivers board intigrated so they only need the control signal comimg from the microcontroller. The motor voltage is supplied from the power management board.

4.4 Robot Hub Board

The robot hub board is the main board that connects all the components together. The board includes the stm32h7 microcontroller that controls the motors and the sensors. The board also acts as a carrier board for the raspberry pi compute module, where different peripherals can be connected to the raspberry pi via the robot hub board. High-density edge connector is used to connect the raspberry pi to the robot hub board. UART is used to communicate between the microcontroller and the raspberry pi.

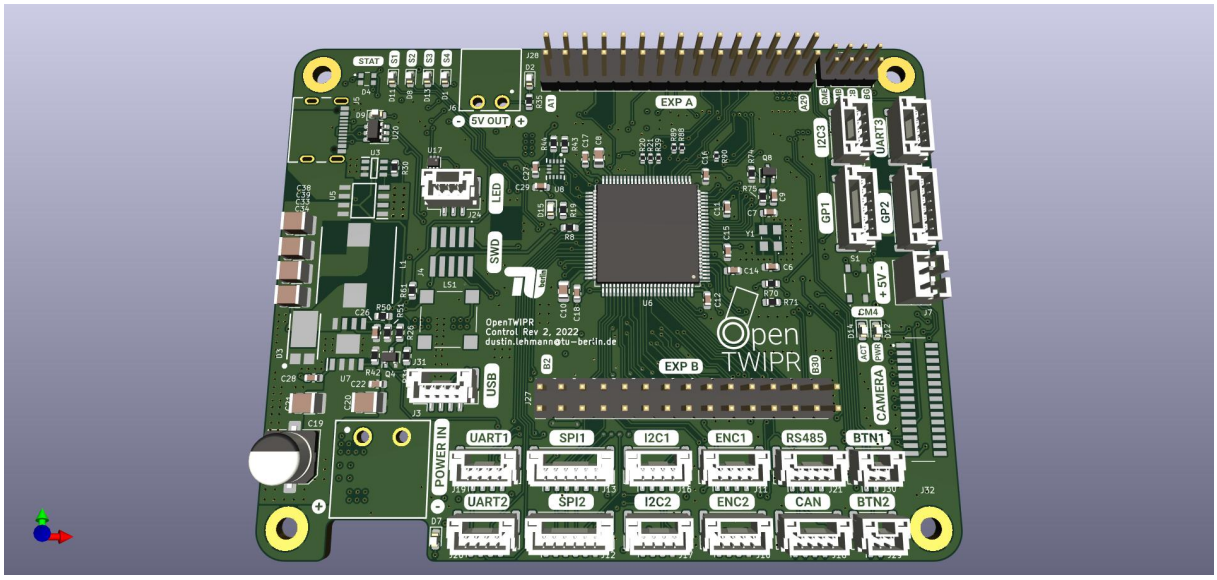


Figure 4.9: Robot Hub Board

4.5 Power Management

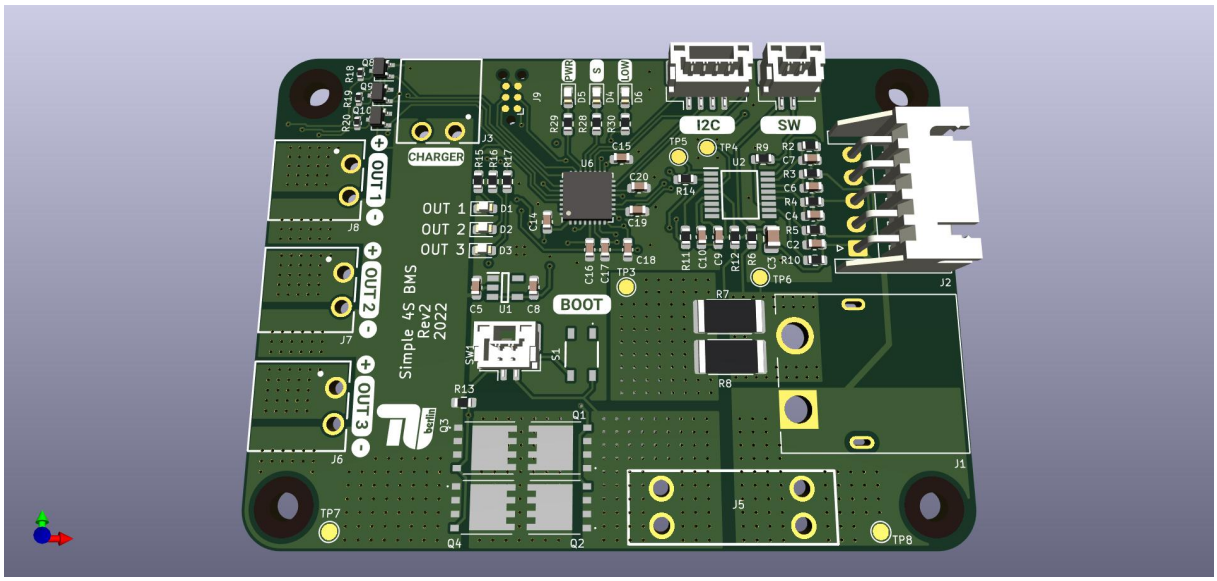


Figure 4.10: Power Management Board

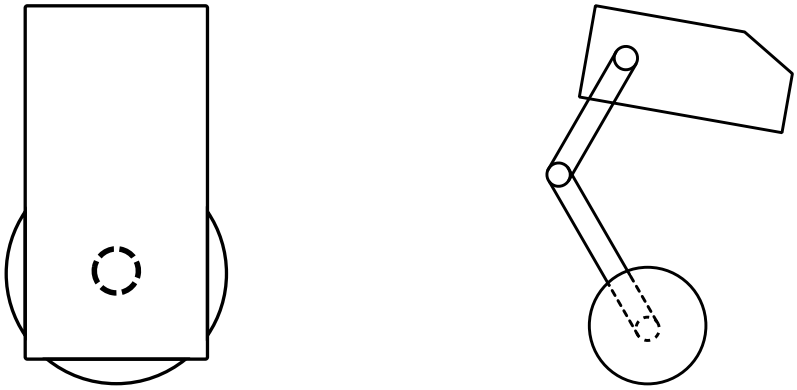
The power mangment board is responsible for managing the power distribution between the different components. The board has a stm32l4 microcontroller that monitors the battery voltage and the current consumption of the motors. The board has a 15 A fuse that protects the battery from over current. The board has a 4-cell battery undervoltage, overvoltage, overcurrent protection. The board has a cell balancing circuit that balances the voltage between the cells of the battery. The board has a high power MOSFET switches that controls the power distribution between the different components.

5 Modelling and Simulation

In this pivotal chapter, we meticulously derive the center of gravity and the moment of inertia for the two-wheeled self-balancing robot. These parameters are the linchpins of our dynamic analysis, serving as the critical variables within the equations of motion that govern the robot's behavior. By calculating these values with precision, we can substitute them into our dynamic equations, thereby tailoring the model to reflect the true dynamics of the robot. This process not only enhances the accuracy of our simulations but also ensures that the control strategies developed are based on a robust and representative model of the robot's physical capabilities. The careful derivation of these parameters is a testament to the thoroughness of our approach, ensuring that the resulting model is both reliable and predictive of the robot's real-world performance.

5.1 Mathematical Modelling

The two-wheeled Inverted Pendulum robot model is as shown in the figure 5.1b where it consists of two legs that include a hip and knee joints as well as wheels at the end of each leg. The robot needs to constantly adjust its posture to be able to maintain the balance, just like how the human being balances when standing on the feet. This new TWIPR model is the latest iteration in the evolution of its predecessor.



(a) Old model **(b)** New model

Figure 5.1: Comparison between the old and new models

5.1.1 2D Dynamics

The scope of this project is limited to 2D dynamics. However, the model can be extended to 3D dynamics and controller synthesis for the 3D dynamics in the future. The 2D dynamics is considered for simplification purposes and to reduce the complexity of the model. The 2D dynamics modeling takes into account the robot's movement in the x-y plane and the rotation of the knee joint, hip joint, and the wheels around the z-axis. The 2D dynamics modeling considers one leg and half of the body mass. The other leg and the other half of the mass are symmetrical to the first leg and the first half of the body.

5.1.2 Center of Gravity

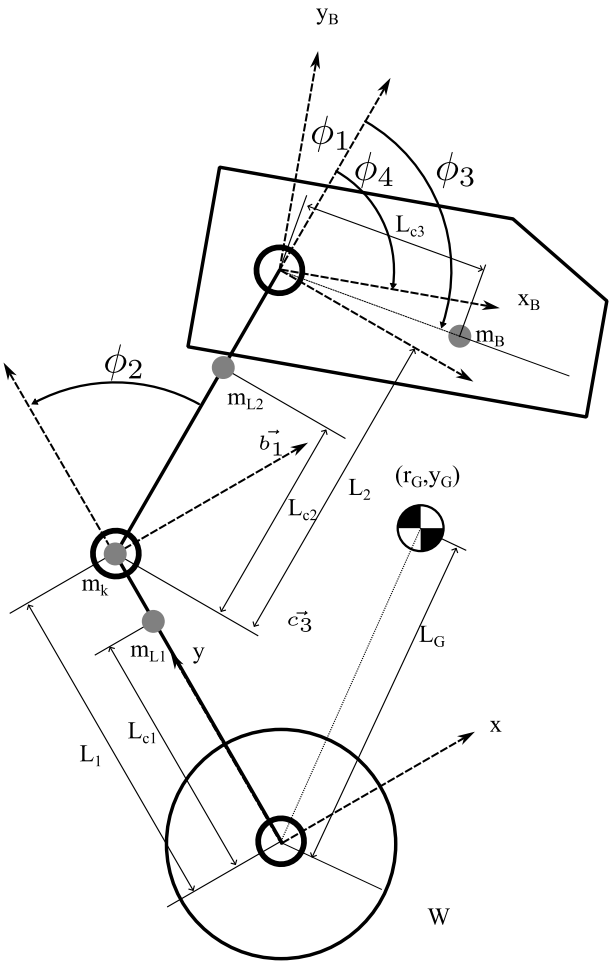


Figure 5.2: Figure illustrating the mechanical model with the local coordinate system to determine the center of gravity.

The calculation of the center of gravity is crucial for the dynamic analysis of the robot. The center of gravity is calculated by taking into account the weights of the links, the knee joint, the body and the distances between these weights and the local coordinate system of the robot. The stability of the robot is directly impacted by the center of gravity position, which also affects how it reacts to forces and moments from the outside world. The following equations are used to calculate the center of gravity location referencing the local coordinate system of the robot as shown in the figure 5.2.

$$x_{CG} = \frac{m_{L1} \cdot 0 + m_K \cdot 0 + m_{L2} \cdot L_{C2} \cdot \sin(\phi_2) + m_B \cdot (L_2 \cdot \sin(\phi_2) + L_{C3} \cdot \sin(\phi_2 + \phi_3))}{m_{L1} + m_{L2} + m_K + m_B} \tag{5.1}$$

Where x_{CG} is the center of gravity in the x-axis.

$$y_{CG} = \frac{m_{L1} \cdot L_{C1} + m_K \cdot L_1 + m_{L2} \cdot (L_1 + L_{C2} \cdot \cos(\phi_2)) + m_B \cdot (L_1 + L_2 \cdot \cos(\phi_2) + L_{C3} \cdot \cos(\phi_2 + \phi_3))}{m_{L1} + m_{L2} + m_K + m_B} \tag{5.2}$$

Where y_{CG} is the center of gravity in the y-axis.

$$L_G = \sqrt{x_{CG}^2 + y_{CG}^2} \tag{5.3}$$

The distance between the center of gravity and the Wheel Joint is calculated using the Pythagorean theorem.

$$\theta = \arctan\left(\frac{x_{CG}}{y_{CG}}\right) \tag{5.4}$$

The angle between the center of gravity L_G and the knee wheel link L_1 is calculated using the arctangent function.

5.1.3 Moment of inertia

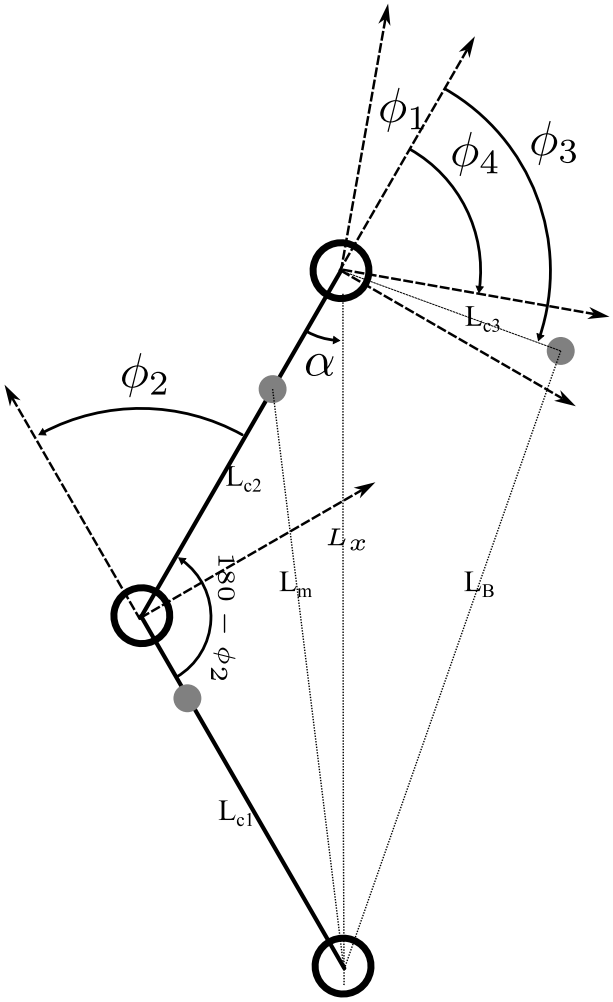


Figure 5.3: Schematic representation detailing the requisite angles and lengths for calculating the moment of inertia.

Moment of inertia calculations are important for the dynamic analysis of the robot. In order to calculate the moment of inertia, the lengths from the wheel joint to the center of mass of each link are calculated. The moment of inertia is calculated using the following equations.

$$L_m = \sqrt{L_1^2 + L_{C2}^2 - L_1 L_{C2} \cos(180 - \phi_2)} \tag{5.5}$$

Where L_m is the distance between the wheel joint and the center of mass of the knee wheel link.

$$L_x = \sqrt{L_1^2 + L_2^2 - L_1 L_2 \cos(180 - \phi_2)} \tag{5.6}$$

$$\alpha = \cos^{-1}\left(\frac{l_2^2 + l_x^2 - l_1^2}{2l_2 l_x}\right) \tag{5.7}$$

$$L_b = \sqrt{L_x^2 + L_{C3}^2 - LxL_{C3}\cos(180 - \alpha - \phi_3)} \tag{5.8}$$

$$I_{L1} = \frac{1}{12} m_{L1}(a_1^2 + b_1^2) \tag{5.9}$$

$$I_{L2} = \frac{1}{12} m_{L2}(a_2^2 + b_2^2) \tag{5.10}$$

$$I_B = \frac{1}{12} m_B(a_B^2 + b_B^2) \tag{5.11}$$

Since the knee wheel link, the hip knee link and the body are assumed to be cuboids, the moment of inertia of each link is calculated using the equation above.

$$I_K = \frac{1}{2} m_K R_m^2 \tag{5.12}$$

The moment of inertia of the knee joint is calculated using the equation above where the knee joint is assumed to be a cylinder.

$$I = I_{L1} + m_{L1}L_{C1}^2 + I_K + m_K L_1^2 + I_{L2} + m_{L2}L_m^2 + I_B + m_B L_b^2 \tag{5.13}$$

The total moment of inertia of the robot is calculated using the equation above.

In order to predict the behavior of the robot under different settings, the modeling procedure entails constructing mathematical representations of the robot's dynamics and control systems.

5.1.4 Assumptions and Parameters

Many assumptions as outlined in 5.2 for the figure 5.2 were made to simplify the model and reduce the complexity of the calculations. The weights and the lengths of the links are constant, and the angles ϕ_2, ϕ_3 are input variables that can be controlled to adjust the robot's posture.

Table 5.1: Parameters of the mechanical system

Parameter	Value	Description
L_1	0.017 m	Length of the Wheel knee Link
L_2	0.017 m	Length of the Body knee Link
L_{C1}	0.013 m	Distance between the Wheel Joint and the center of mass of the Wheel knee Link
L_{C2}	0.013 m	Distance between the Knee Joint and the center of mass of the Body knee Link
L_{C3}	0.01 m	Distance between the Hip Joint and the center of mass of the body
m_{L1}	0.2 kg	Mass of the Wheel knee Link
m_{L2}	0.2 kg	Mass of the Body knee Link
m_K	0.2 kg	Mass of the Knee Joint
m_B	0.5 kg	Mass of the Body
L_G	-	Distance between the center of gravity and the Wheel Joint
θ	-	Angle between the center of gravity and the Wheel knee Link
ϕ_1	-	Angle between the Body knee Link and the y_B vertical axis of the body
ϕ_2	-	Angle between the Wheel knee Link and the Body knee Link
ϕ_3	-	Angle between the Body knee Link and the center of mass of the body
ϕ_4	-	Angle between the Body knee Link and the X_B horizontal axis of the body
L_m	-	Distance between the Wheel Joint and the center of mass of the Wheel knee Link
L_x	-	Distance between the Wheel Joint and the center of mass of the hip knee Link
L_b	-	Distance between the Wheel Joint and the center of mass of the body
α	-	Angle between L_x and L_2
I_{L1}	-	Moment of inertia of the knee wheel link around the z-axis
I_{L2}	-	Moment of inertia of the hip knee link around the z-axis
I_K	-	Moment of inertia of the knee joint around the z-axis
I_B	-	Moment of inertia of the body around the z-axis
I	-	Total moment of inertia of the robot around the z-axis
a_1	-	Length of the knee wheel link
b_1	-	Width of the knee wheel link
a_2	-	Length of the hip knee link
b_2	-	Width of the hip knee link
a_B	-	Length of the body
b_B	-	Width of the body
R_m	-	Radius of the knee joint
a_B	-	Length of the body
b_B	-	Width of the body

5.1.5 Dynamics of the Two-Wheeled Inverted Pendulum Robot

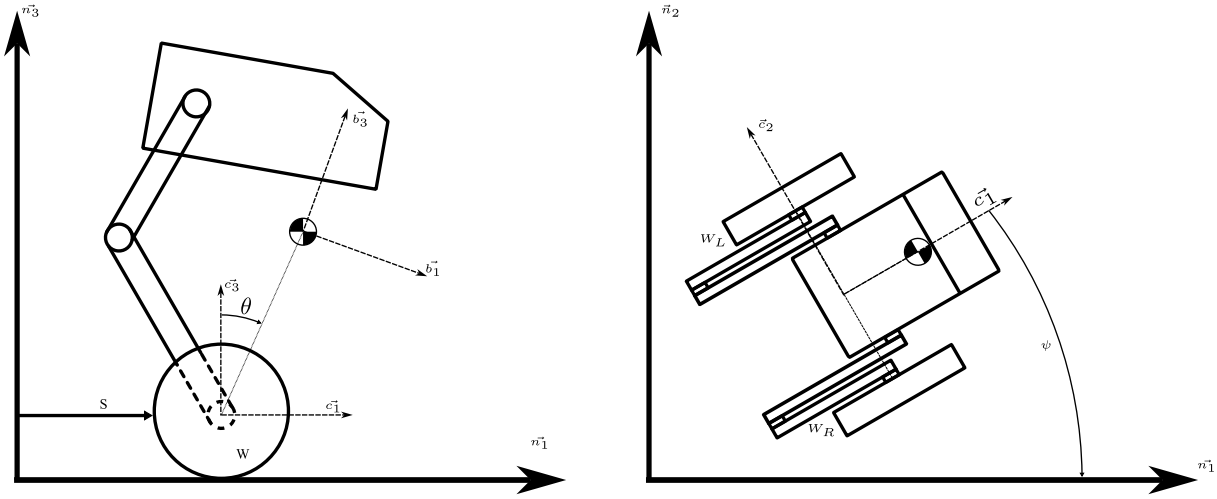


Figure 5.4: Figure illustrating the Two-Wheeled Inverted Pendulum Robot Dynamics.

Given the functions $B_i : \mathbb{R} \to \mathbb{R}$, $C_{ij} : \mathbb{R} \to \mathbb{R}$, $D_{ij} : \mathbb{R} \to \mathbb{R}$, and $V_i : \mathbb{R} \to \mathbb{R}$, $i,j \in \{1,2,3\}$, the equations of motion are given by:

$$\ddot{s} = \frac{\sin(\theta)}{V_1(\theta)} \left(-C_{11}(\theta)g + C_{12}\dot{\theta}^2 + C_{13}(\theta)\dot{\psi}^2\right) - \frac{D_{11}(\theta)}{V_1(\theta)}\dot{s} + \frac{D_{12}(\theta)}{V_1(\theta)}\dot{\theta} + \frac{B_1(\theta)}{V_1(\theta)}(\tau_L + \tau_R) \tag{5.14}$$

$$\ddot{\theta} = \frac{\sin(\theta)}{V_1(\theta)} \left(C_{21} - C_{22}(\theta)\dot{\theta}^2 - C_{23}(\theta)\dot{\psi}^2\right) + \frac{D_{21}(\theta)}{V_1(\theta)}\dot{s} - \frac{D_{22}(\theta)}{V_1(\theta)}\dot{\theta} - \frac{B_2(\theta)}{V_1(\theta)}(\tau_L + \tau_R) \tag{5.15}$$

$$\ddot{\psi} = \frac{\sin(\theta)}{V_2(\theta)} \left(C_{31}(\theta)\dot{\theta}\dot{\psi} - C_{32}(\theta)\dot{\psi}\dot{s}\right) - \frac{D_{33}(\theta)}{V_2(\theta)}\dot{\psi} - \frac{B_3}{V_2(\theta)}(\tau_L - \tau_R) \tag{5.16}$$

The equations of motion are derived in [15]. The functions $B_i : \mathbb{R} \to \mathbb{R}$, $C_{ij} : \mathbb{R} \to \mathbb{R}$, $D_{ij} : \mathbb{R} \to \mathbb{R}$, and $V_i : \mathbb{R} \to \mathbb{R}$, $i,j \in \{1,2,3\}$, are given by:

$$C_{11}(\theta) = m_B^2 l^2 \cos(\theta) g, \tag{5.17}$$

$$C_{12} = (I_2 + m_B l^2) m_B l, \tag{5.18}$$

$$C_{13}(\theta) = (I_2 + m_B l^2) m_B l + m_B l (I_3 - I_1 - m_B l^2) \cos^2(\theta), \tag{5.19}$$

$$C_{21} = (m_B + 2m_W + \frac{2J}{r^2}) m_B l, \tag{5.20}$$

$$C_{22}(\theta) = m_B^2 l^2 \cos(\theta), \tag{5.21}$$

$$C_{23}(\theta) = (m_B^2 l^2 + (m_B + 2m_W + \frac{2J}{r^2})(I_3 - I_1 - m_B l^2)) \cos(\theta). \tag{5.22}$$

$$C_{31}(\theta) = 2(I_3 - I_1 - m_B l^2) \cos(\theta), \tag{5.23}$$

$$C_{31}(\theta) = m_B l, \tag{5.24}$$

$$D_{11}(\theta) = \frac{(I_2 + m_B l^2) 2 c_\alpha}{r^2} - \frac{m_B l \cos(\theta) 2 c_\alpha}{r}, \tag{5.25}$$

$$D_{12}(\theta) = \frac{(I_2 + m_B l^2) 2 c_\alpha}{r} - 2 m_B l \cos(\theta) c_\alpha, \tag{5.26}$$

$$D_{21}(\theta) = \frac{(m_B + 2m_W + \frac{2J}{r^2})2c_\alpha}{r} + \frac{m_B l\cos(\theta)2c_\alpha}{r^2}, \tag{5.27}$$

$$D_{22}(\theta) = \frac{(m_B + 2m_W + \frac{2J}{r})2c_\alpha + m_B l\cos(\theta)2c_\alpha}{r}, \tag{5.28}$$

$$D_{33}(\theta) = \frac{d^2 c_\alpha}{2r^2}, \tag{5.29}$$

$$B_1 = (I_2 + m_B l^2)\frac{1}{r} + m_B l\cos(\theta), \tag{5.30}$$

$$B_2 = \frac{m_B l\cos(\theta)}{r} + m_B + 2m_W + \frac{2J}{r^2}, \tag{5.31}$$

$$B_3 = \frac{d}{2r}, \tag{5.32}$$

$$V_1 = (m_B + 2m_W + \frac{2J}{r^2})(I_2 + m_B l^2) - m_B^2 l^2\cos^2(\theta), \tag{5.33}$$

$$V_2 = I_3 + 2K + (m_W + \frac{J}{r^2})\frac{d^2}{2} - (I_3 - I_1 - m_B l^2)\sin^2(\theta). \tag{5.34}$$

The functions $B_i : \mathbb{R} \to \mathbb{R}$, $C_{ij} : \mathbb{R} \to \mathbb{R}$, $D_{ij} : \mathbb{R} \to \mathbb{R}$, and $V_i : \mathbb{R} \to \mathbb{R}$, $i,j \in \{1,2,3\}$, depends on the parameters in table 5.2.

Table 5.2: Parameters of the mechanical system

Parameter	Value	Description
m_B	0.5 kg	mass of the pendulum body
m_W	0.636 kg	mass of a wheel
l	-	distance between the wheel axis and the pendulum's center of gravity
d	-	distance between the two wheels
J	$5.175e^{-4}$ kgm^2	moment of inertia of a wheel w.r.t. Reference frame {C} in direction of c_2
K	-	moment of inertia of a wheel w.r.t. reference frame {C} in direction of c_3
I_1	-	moment of inertia of pendulum's body w.r.t. Reference frame {B} in direction of b_1
I_2	-	moment of inertia of pendulum's body w.r.t. Reference frame {B} in direction of b_2
I_3	-	moment of inertia of pendulum's body w.r.t. Reference frame {B} in direction of b_3
c_α	$4.630e^{-4}$ Nms	viscous friction coefficient

5.1.6 Linearization of the Two-Wheeled Inverted Pendulum Robot

In this section, we linearize the equations of motion of the legged TWIPR robot about the upright equilibrium point. Movement of the robot is restricted to the x-y plane.

The torque inputs are τ_L and τ_R for the left and right wheels, respectively.

$$\tau_L = \tau_R = \tau \tag{5.35}$$

the sum of the torques τ_L and τ_R is equal to the combined torque input u which is an input and a variable.

$$u = \tau_L + \tau_R = 2\tau \tag{5.36}$$

The degree of freedom for the rotation around the z-axis ψ is eleinated by assuming that the robot is always moving in the x-y plane in a straight line. therfore the equation of motion would be as follows:

$$\ddot{s} = \frac{\sin(\theta)}{V_1(\theta)}\left(-C_{11}(\theta)g + C_{12}\dot{\theta}^2\right) - \frac{D_{11}(\theta)}{V_1(\theta)}\dot{s} + \frac{D_{12}(\theta)}{V_1(\theta)}\dot{\theta} + \frac{B_1(\theta)}{V_1(\theta)}u \tag{5.37}$$

$$\ddot{\theta} = \frac{\sin(\theta)}{V_1(\theta)} \left(C_{21} - C_{22}(\theta)\dot{\theta}^2\right) + \frac{D_{21}(\theta)}{V_1(\theta)}\dot{s} - \frac{D_{22}(\theta)}{V_1(\theta)}\dot{\theta} - \frac{B_2(\theta)}{V_1(\theta)}u \tag{5.38}$$

Therefore the state vector would be as follows:

$$x = \begin{bmatrix} \theta \\ \dot{\theta} \\ s \\ \dot{s} \end{bmatrix} \tag{5.39}$$

$$f_1(x) = \frac{\sin(\theta)}{V_1(\theta)} \left(C_{21} - C_{22}(\theta)\dot{\theta}^2\right) + \frac{D_{21}(\theta)}{V_1(\theta)}\dot{s} - \frac{D_{22}(\theta)}{V_1(\theta)}\dot{\theta} \tag{5.40}$$

$$f_2(x) = \frac{\sin(\theta)}{V_1(\theta)} \left(-C_{11}(\theta)g + C_{12}\dot{\theta}^2\right) - \frac{D_{11}(\theta)}{V_1(\theta)}\dot{s} + \frac{D_{12}(\theta)}{V_1(\theta)}\dot{\theta} \tag{5.41}$$

$$g_1(x,u) = \frac{B_2(\theta)}{V_1(\theta)}u \tag{5.42}$$

$$g_2(x,u) = \frac{B_1(\theta)}{V_1(\theta)}u \tag{5.43}$$

The system can be represented in the following form:

$$\dot{x} = \begin{bmatrix} x_2 \\ f_1(x) \\ x_4 \\ f_2(x) \end{bmatrix} + \begin{bmatrix} 0 \\ g_1(x,u) \\ 0 \\ g_2(x,u) \end{bmatrix} \tag{5.44}$$

Linearization at the point $x = 0$, which corresponds to the upright equilibrium of the inverted pendulum, simplifies the system's dynamics as follows:

$$\dot{x} = A_c x + B_c u, \tag{5.45}$$

where $A_c \in \mathbb{R}^{4\times4}$ is the system matrix of the time-continuous system, and $B_c \in \mathbb{R}^{4\times1}$ is the input matrix of the time-continuous system. Discretization of the system with a sampling frequency of 50 hertz yields:

$$x_{k+1} = Ax_k + Bu_k, \tag{5.46}$$

where $k \in \mathbb{N}$ indexes the discrete time steps, $x_k \in \mathbb{R}^4$ denotes the state vector at the k-th sample, $u_k \in \mathbb{R}$ denotes the input at the k-th sample, $A \in \mathbb{R}^{4\times4}$ is the system matrix of the time-discrete system, and $B \in \mathbb{R}^{4\times1}$ is the input matrix of the time-discrete system [23].

5.2 Numerical Simulation

The discrete numerical simulation is a computational method that is used to model the behavior of dynamic systems over discrete periods of time intervals. In contrast to the continuous numerical simulation, the discrete numerical simulation that typically uses the Euler method to solve the differential equations analytically or working with infinitesimally small-timescale numerical solvers.

5.2.1 Discrete numerical simulation

Discrete numerical simulations approximate the state of the system step by step and evaluate the behavior of the system at predetermined intervals. When an analytical solution is not feasible or real-time simulation is required due to the complexity of the system, this approach is particularly helpful.

5.2.2 Discrete double integration method

The discrete double integration method is used to calculate the state vector of the system at each time step. The state vector is calculated by integrating the acceleration twice to get the position and velocity. The state vector is calculated using the following equation:

$$x_{k+1} = x_k + \dot{x_k}\Delta t + \frac{1}{2}\ddot{x_k}\Delta t^2 \tag{5.47}$$

where x_k is the state vector at the k-th sample, $\dot{x_k}$ is the derivative of the state vector at the k-th sample, $\ddot{x_k}$ is the second derivative of the state vector at the k-th sample, and Δt is the time step.

5.3 Simulation Environment

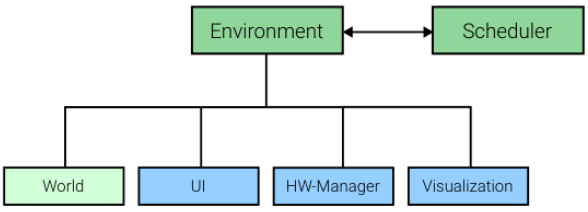


Figure 5.5: The environment consists always of a world entity, other parts like user in- terface(UI), HW-manager and visualization are optional, it is closely linked to the Scheduler [7]

The simulation environment is a software framework that is used to simulate the dynamics of the robot. The simulation enviroment is a A complex virtual environment created for realistic testing is the simulation environment. It comprises a variety of spaces with distinct states, dimensions, and mapping capabilities that enable the creation of a wide range of scenarios [7].

5.3.1 Physics and Object Primitives

The simulation ensures realistic movements and interactions by incorporating intricate physics. It makes use of object primitives such as spheres, cylinders, and cuboids, which serve as a basis for building a variety of structures and objects[7].

5.3.2 Collision Checking and Dynamics

The simulation uses sophisticated collision checking techniques, such as pre-checking and distinct algorithms for various situations, such as collisions involving cuboids. To simulate realistic physical interactions between objects, this feature is essential[7].

5.3.3 Objects and Agents

A range of objects, both static and dynamic, including agents that imitate autonomous behavior, are present in the environment. This variety improves the simulation's ability to mimic interactions in the real world.

5.3.4 Scheduling and Real Robot Integration

An advanced scheduling system controls the order and timing of events. Furthermore, the simulation incorporates real robots in a unique way, combining virtual and physical components for thorough testing[7].

5.3.5 Integration of the Model

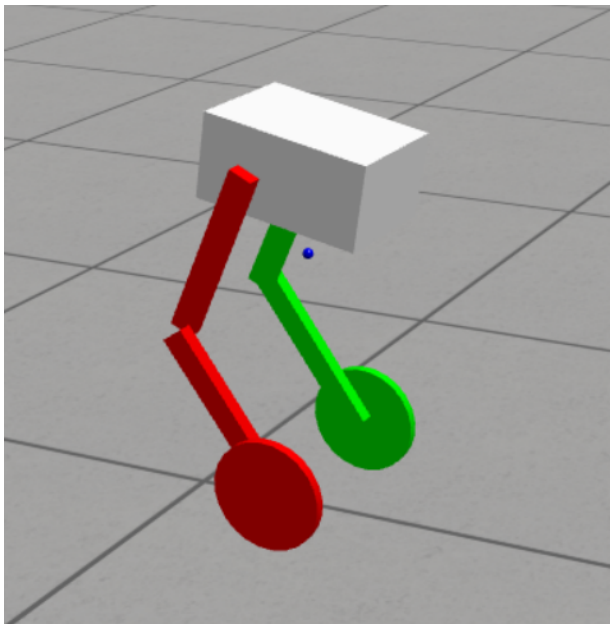


Figure 5.6: The simulation environment

As shown in the figure 5.6, the simulation environment simulates the dynamics of the robot and the controller. where the blue sphere represents the robot center of gravity which changes its location according to the robot's dynamics.

5.4 Controller Synthesis

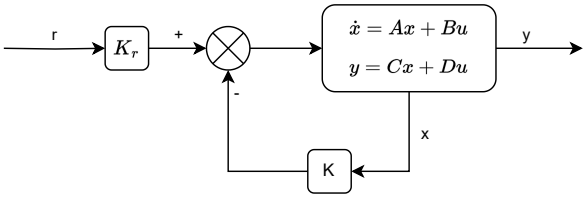


Figure 5.7: Block diagram of a state feedback controller

In the realm of modern control theory, State-Space controllers are a pivotal component for managing the behavior of complex systems.Among the most prominent of these controllers are the Linear Quadratic Regulator (LQR) and Pole-Placement controllers, both of which are robust and effective in various applications. This section provides an overview of these controllers and their application to the legged TWIPR robot.

5.4.1 Linear Quadratic Regulator (LQR)

The Linear Quadratic Regulator (LQR) is a state-space controller that uses a quadratic cost function to determine the optimal control input for a given system. It aims to minimize the cost function by adjusting the control input, thereby ensuring that the system's state converges to the desired state. The LQR controller is defined by the following equation:

$$u(t) = -Kx(t) \tag{5.48}$$

where $u(t)$ is the control input, $x(t)$ is the state vector, and K is the gain matrix. The gain matrix is calculated using the following equation:

$$K = R^{-1}B^T P \tag{5.49}$$

where R is the control weight matrix, B is the input matrix, and P is the solution to the Riccati equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{5.50}$$

where A is the state matrix and Q is the state weight matrix. The state matrix, state weight matrix, and control weight matrix are defined as follows:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.51}$$

$$R = \begin{bmatrix} 1 \end{bmatrix} \tag{5.52}$$

The LQR controller is implemented in the simulation environment using the *Python Control Systems Library* [24].

5.4.2 Pole-Placement Technique

The Pole-Placement technique is a state-space controller that uses the Ackermann formula to determine the optimal control input for a given system. It aims to place the poles of the system at the desired locations by adjusting the control input, thereby ensuring that the system's state converges to the desired state.placing the poles is not initiutive for high order systems or systems with multiple actuators. The Pole-Placement controller is defined by the following equation:

$$u(t) = -Kx(t) \tag{5.53}$$

where $u(t)$ is the control input, $x(t)$ is the state vector, and K is the gain matrix. The gain matrix is calculated using the following equation:

$$K = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix} \tag{5.54}$$

where k_1, k_2, and k_3 are the gains for the first, second, and third states, respectively. The gains are calculated using the following equation:

$$k_i = \frac{1}{b_i}\left(\sum_{j=0}^{n-1} a_{n-j}\alpha_{i+j} - \alpha_i\right) \tag{5.55}$$

where a_i is the coefficient of the characteristic polynomial, b_i is the coefficient of the denominator polynomial, and α_i is the desired location of the ith pole. The characteristic polynomial and denominator polynomial are defined as follows:

$$a_i = \begin{cases} 1 & i = 0 \\ 0 & i \neq 0 \end{cases} \tag{5.56}$$

$$b_i = \begin{cases} 1 & i = 0 \\ 0 & i \neq 0 \end{cases} \tag{5.57}$$

The Pole-Placement controller is implemented in the simulation environment using the *Python Control Systems Library* [24].

5.4.3 Configuration Specificity

These controllers are specific to one configuration for the knee angle and hip angle of the robot. Retuning the controllers is required when the configuration changes.

```
1    def change_knee_angle(self):
2        self.agent1.set_leg_angles(hip_angle=self.agent1.dynamics.model.hip_angle,
3                                   knee_angle=self.agent1.dynamics.model.knee_angle + deg2rad
     (5))
4
5    def change_hip_angle(self):
6        self.agent1.set_leg_angles(hip_angle=self.agent1.dynamics.model.hip_angle + deg2rad(5)
     ,
7                                   knee_angle=self.agent1.dynamics.model.knee_angle)
```

Lst. 5.1: Changing the knee and hip angles of the robot

where The code 5.1 Shows the functions that changes the knee and hip angles of the robot.

```
1    def set_leg_angles(self, knee_angle: float, hip_angle: float):
2        self.dynamics.model.knee_angle = knee_angle
3        self.dynamics.model.hip_angle = hip_angle
4        self.linear_dynamics = TWIPR_3D_Linear(self.dynamics.model, Ts, self.poles, self.
     eigenvectors)
5        self.state_ctrl_K = np.hstack((np.zeros((2, 1)), self.linear_dynamics.K))
```

Lst. 5.2: Changing the knee and hip angles in the dynamics and retuning the controller

In the above code 5.2 It hows the function that changes the knee and hip angles in the dynamics and retunes the controller.

5.5 Simulation Analysis

The simulation analysis is a process of analyzing the performance of the controllers by conducting a series of simulations to evaluate their responses to various inputs in different configurations.

5.5.1 Controller Responses

In order to analyze the performance of the controllers, we conducted a series of simulations to evaluate their responses to various inputs.

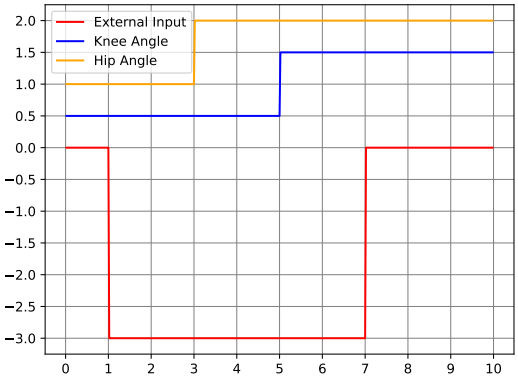
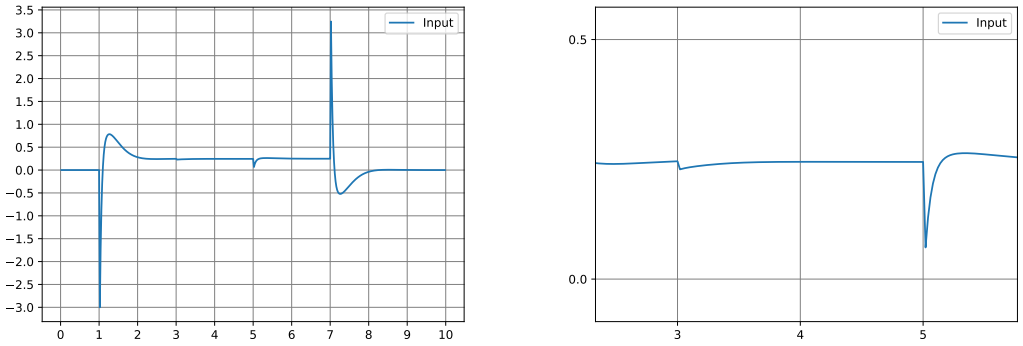


Figure 5.8: Test input

As shown in the figure 5.8, this test input data is selected to evaluate the performance of the controllers. External control input is applied at $t = 1$ and is maintained until $t = 7$ to see the response. Additionally, changing the knee and hip angles at $t = 3$ and $t = 5$ respectively to see the response of the controller to the change in the configuration.



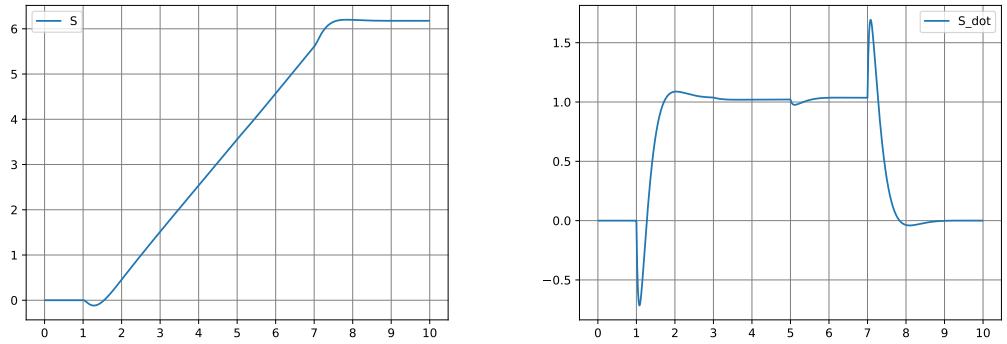
(a) Step response of the internal input

(b) Step response of the hip and knee angles

Figure 5.9: Step response of the internal input

The internal input is the input that is generated by the controller and is used to control the robot. The step response of the internal input is shown in the figure 5.9.The figure shows the response at time $t = 1$ when the external input is applied where it overshoots and then converges to a value of 0.25 which indicates that the robot is moving forward and the external input is still applied. The figure 5.9b shows the response of the hip and knee angles to the change in the configuration at $t = 3$ and $t = 5$ respectively.

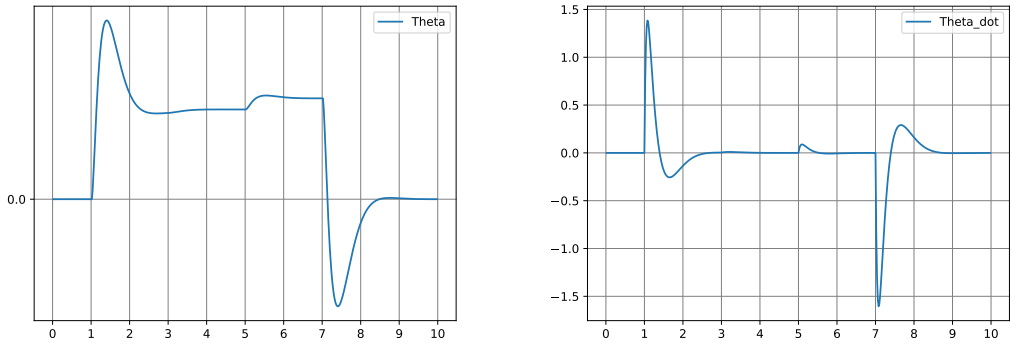
The effect of the change in knee angle is more significant than the effect of the change in the hip angle since the knee angle has much higher effect on the center of gravity location than the hip angle.The stop of the external input is reflected in the response at $t = 7$ where the internal input converges to 0 which indicates that the robot is not moving forward anymore.



(a) S **(b)** S dot

Figure 5.10: S and S dot

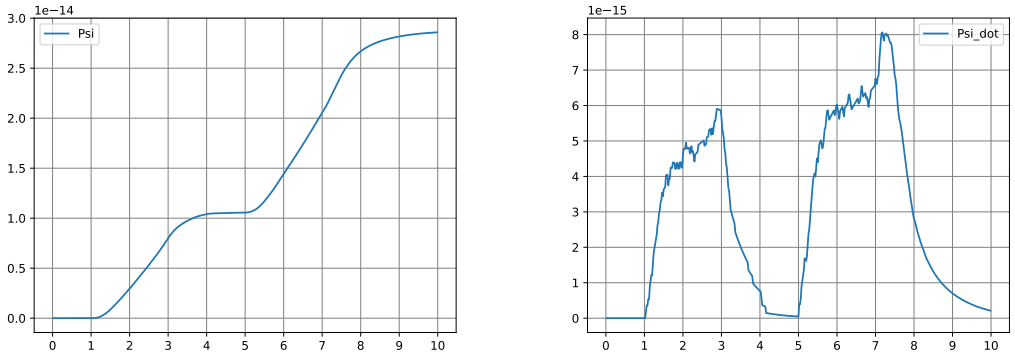
As shown in the figure 5.10, the s is the x position of the robot. The s dot is the derivative of the s. As showen in the figure 5.10a at $t = 1$ the robot starts moving slightly backward befor it starts moving forward. The center of gravity is learned a bit forward with translates into a slightly backward movement.Afterwards the controller keeps the balance of the robot while it is moving forward.The external input is stopped at $t = 7$ which is reflected in the figure 5.10a at $t = 7$ where the robot comes to a stop. The figure 5.10b shows the velocity of the robot. The velocity is disrupted at $t = 3$ and $t = 5$ when the hip and knee angles are changed respectively.



(a) Theta **(b)** Theta dot

Figure 5.11: Theta and Theta dot

As shown in the figure 5.11, the theta is the angle between the center of gravity and the vertical axis to the ground. The theta dot is the derivative of the theta.



(a) Psi **(b)** Psi dot

Figure 5.12: Psi and Psi dot

As shown in the figure 5.12, the

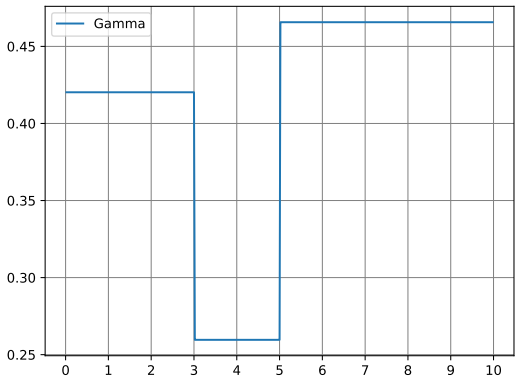


Figure 5.13: Gamma

The Gamma angle is the angle between the center of gravity and the link 1. As shown in the figure 5.13, the Gamma angle is changed at $t = 3$ and $t = 5$ when the hip and knee angles are changed respectively. It decreased to zero if the robot is verticallly aligned with the ground meaning when the hip and knee angles are zero.

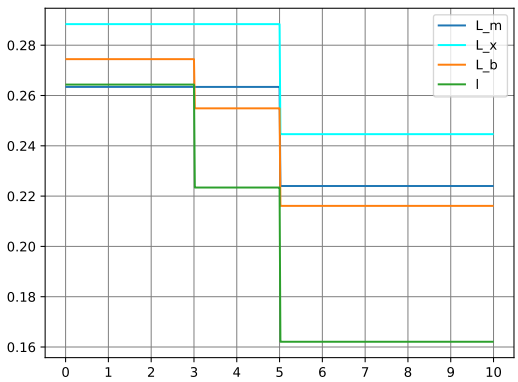


Figure 5.14: Lm Lx Lb and L

As shown in the figure 5.14, the Lm Lx Lb and L are changed at $t = 3$ and $t = 5$ when the hip and knee angles are changed respectively.

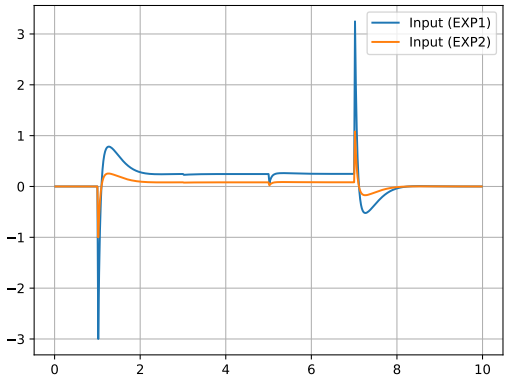


Figure 5.15: Comparison between the two external inputs

In the figure 5.15, two different external inputs are applied to the robot. In experment 1, 1 unit of external input is applied to the robot. In experment 2, 3 units of external input is applied to the robot. The experment 2 shows a better response than experment 1.The overshoot is less and the robot converges to the desired state faster. In addition, the robot is more stable when the hip and knee angles are changed at $t = 3$ and $t = 5$ respectively.

5.5.2 Impact of Non-Retuning

If the controller is not returned when the configuration changes, the old K gains that rely on the old center of gravity location will be used, resulting in a poor response.

5.5.3 Influence of Leg Configuration

5.5.4 Controller Setting Comparisons

6 Mechanical Assembly

In this chapter, the mechanical assembly of the robot is discussed. The mechanical assembly is the process of putting together the mechanical components of the robot. The mechanical components include the chassis which is composed of several parts, the wheels, the motors. The assembly process is described in detail, including the tools and techniques used. The integration of the mechanical and electronic systems is also discussed. The challenges faced during the assembly process are described, along with the troubleshooting and problem-solving strategies employed. Finally, the safety considerations taken during the assembly process are discussed. , the wheels, the motors.

6.1 Components Overview

As shown in the mechanical design chapter, the robot is composed of several mechanical components.The body, hip knee links, knee wheel links, board mounting rack, Wheel motor cover, body cover and body face shield. These components were printed using a 3D printer with different configuration for each part.Other components such as the wheels, motors, motor shaft hub, screws and thread inserts were bought from the market.

6.2 Fabrication

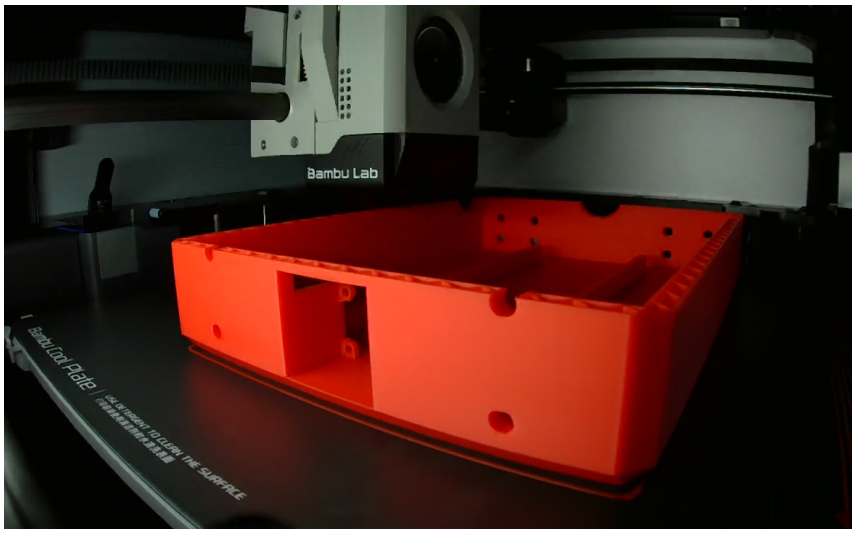


Figure 6.1: Printing the body

The print infill, orientation material and layer height were changed to suit the part. Fifty percent infill was used for the body, and seventy percent infill was used for links to make them strong enough to withstand the forces applied on them and to avoid breaking at weak points. The orientation of the parts was changed to make the print stronger and to avoid the need for support material. The aim was to make the printed layers perpendicular to the forces applied on the part. PLA was used as the printing material for all the parts except for the body face shield which was printed using TPU to make it flexible. Different

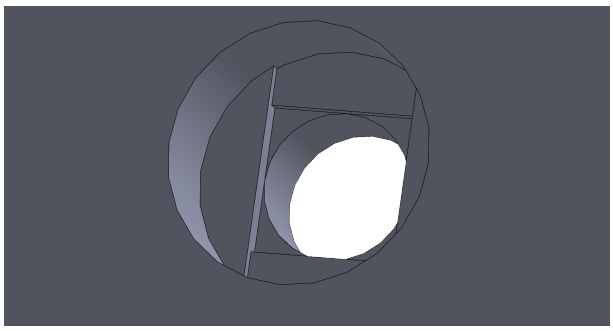


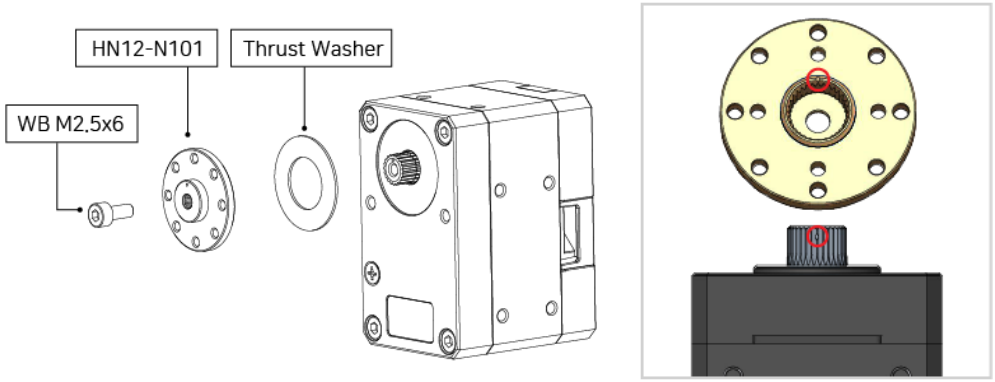
Figure 6.2: Design techniques used to avoid the need for support material

design techniques as shown in figure 6.2were used specially for the place where the socket screws heads are inserted to avoid the need for support material.

6.3 Assembly Process

6.3.1 Motors Assembly

The hip and knee motors required assembly before they could be mounted on the chassis.



(a) Hip and knee motors assembly **(b)** Horn assembly marking

Figure 6.3: Comparison between the hip and knee motors assembly and the horn assembly marking

The assembly process is shown in figure 6.3a shows the normal horn assembly. the thrust washer should be placed between the horn and the motor to avoid friction between the horn and the motor. the horn should be tightened using the screw.As shown in figure 6.3b the indexing mark on the output horn is aligned with the index marking on the output shaft.

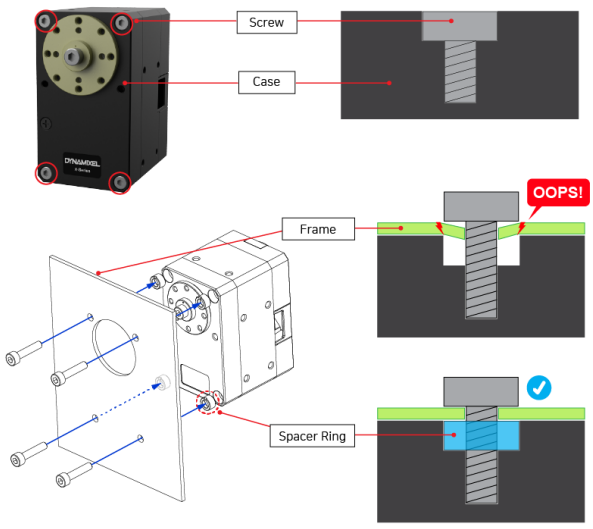


Figure 6.4: Motor spacer ring

The motor spacer ring shown in figure 6.4 was used to make to fill the gap between the motor case and chassis frame. The frame thickness was added to the length of the new screw to make sure that the screw is not larger than the depth of the mounting point or the motor case may be damaged.

6.3.2 Screws and thread inserts

The screws and thread inserts used in the assembly are shown in the table 6.1. The precise dimensions of the screws and thread inserts were important to make sure that the assembly process goes smoothly. The tolerance of 0.1 mm for the screws was acceptable for the assembly process except for the screws used for the hip and knee motors horns as it could touch the thrust washer and cause damage to the motor.

Table 6.1: Screws and thread inserts used in the assembly

Component	Quantity	Screw size
Wheel Motor	8	M3*8
Knee Motor Horn	16	M2*13
Knee Motor front	8	M2.5*14
Knee Motor Top	8	M2.5*5
Knee Motor sides	20	M2.5*5 or M2.5*5 .5
Hip Motor front	8	M2.5*6
Hip Motor Horn	16	M2*15
Hip Motor side	8	M2.5*15
Body	12	M2.5*6
Wheel Motor cover	20	M2.5*6
Thread inserts	36	M2.5 x 5.7

6.3.3 Chassis Assembly

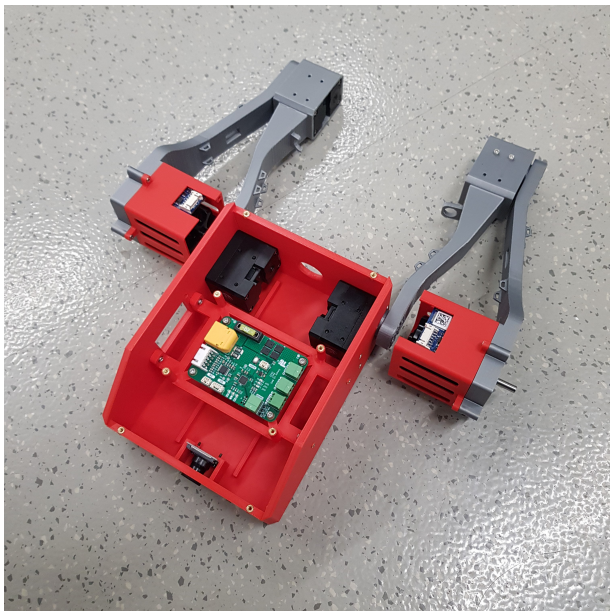
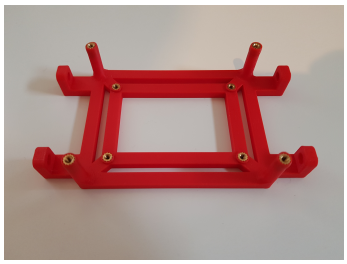


Figure 6.5: Assembled chassis

The assembly process started with the chassis, the chassis was assembled using screws and thread inserts. First the thread inserts were inserted into the chassis using soldering iron, then the screws were used to assemble the motors to the knee wheel links, the hip knee links, the body. then the knee wheel links were assembled to the hip knee links

(a) Body thread inserts

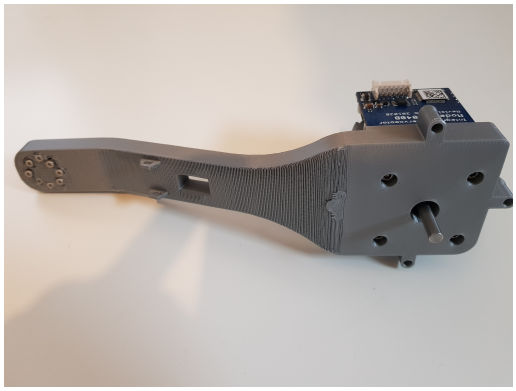
(b) Knee wheel link thread inserts



(c) Board mounting rack thread inserts

Figure 6.6: Thread inserts

The thread inserts were inserted into the chassis using a soldering iron as shown in figure 6.6 for the body, knee wheel links and the board mounting rack. Minimum force was applied to the soldering iron to align the thread inserts surface with the surface of the chassis.

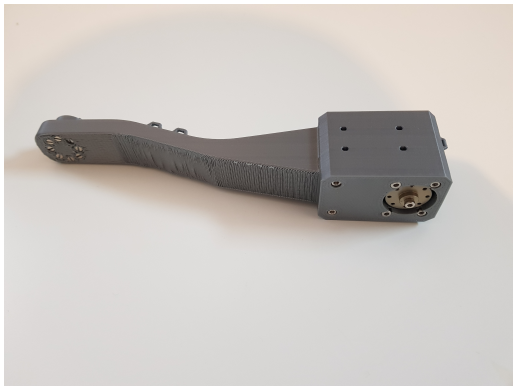


(a) Mounting motors on knee wheel links perspective view

(b) Mounting motors on knee wheel links top view

Figure 6.7: Mounting motors on knee wheel links

The wheel motors were mounted on the knee wheel links using four screws, as shown in figure **??**. The SC040B aluminum extrusion that holds the motor and electronics allows the motor to be mounted on the knee wheel links. free moving air around the motor is important to keep the motor cool and to avoid overheating.



(a) Mounting motors on hip knee links perspective view

(b) Mounting motors on hip knee links top view

Figure 6.8: Mounting motors on hip knee links

After the XM430-W350 is prepared, the motor is mounted into the dedicated cover as shown in figure **??** using 6 main screws, and it can be fixed with extra 10 screws around the cover.

(a) Assembling knee wheel links and hip knee links prespective view

(b) Assembling knee wheel links and hip knee links top view

Figure 6.9: Assembling knee wheel links and hip knee links

The knee wheel links were assembled to the hip knee links using sixteen screws as shown in figure **??** to the Knee motor horns after the motors were mounted in the hip knee links. The screws weren't over tightened to avoid exiting the horn from the other side and touching the motor thrust washer.

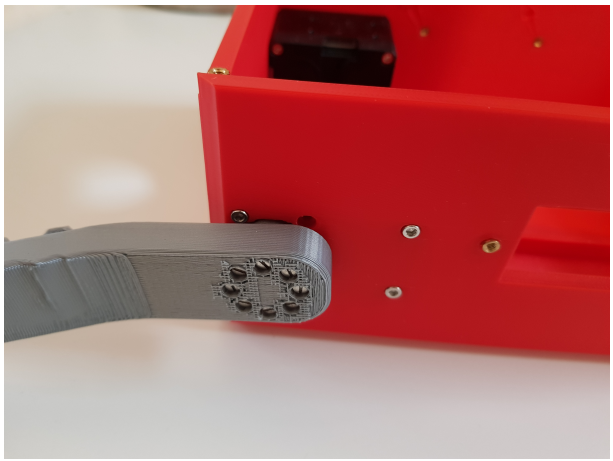


Figure 6.10: Assembling body to hip knee links

The hip knee links were assembled to the body using sixteen screws as shown in figure 6.10 to the hip motor horns after the motors were mounted in the body.

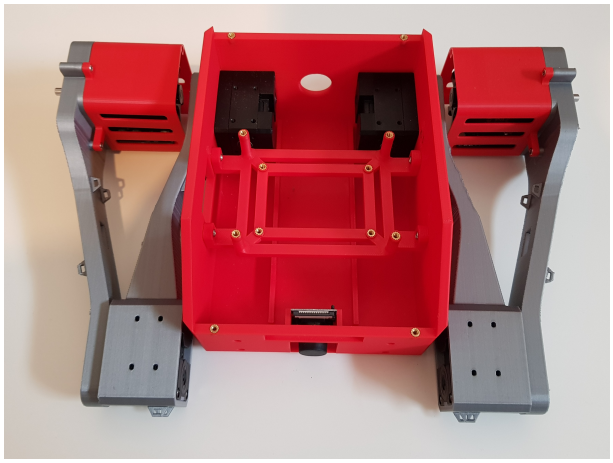


Figure 6.11: Fully assembled chassis

The fully assembled chassis is shown in figure 6.11.In that figure only the motors and the camera are mounted on the chassis.

6.4 Integration of Mechanical and Electronic Systems

6.4.1 Electronics Assembly

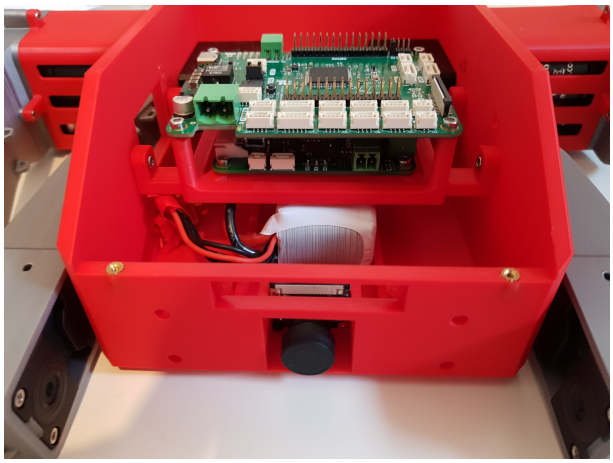


Figure 6.12: Electronics mounted on board mounting rack

The electronics were mounted on the board mounting rack as shown in figure 6.12.The power mangement board was mounted on the board mounting rack first using four screws, then the robot hub board was mounted on top of the power management board using four screws.The battery was placed under the board mounting rack in the space between the two designed brackets to hold it in place.

6.4.2 Wiring tree

According to the wiring tree in the electrical design chapter, the wiring tree was made to fit the lengthes between the components. The wires were cut to the required length and the connectors were crimped to the wires. The wires were connected to the components according to the wiring tree. The wires were tied together using cable ties to make sure they are managed properly and to avoid them interfering with the robot movement. The wires were also tied to the chassis using the designed cable management features to ensure that the wires are fixed in place.

6.5 Troubleshooting and Problem Solving

Different problems were faced during the assembly process. The knee motor was touching the screw so the screw was shortened to avoid damaging the motor. The knee motor didn't fit in the link, so the link was initialy filed to make the motor fit, but then it was reprinted with new dimensions to make the motor fit. Some cabe management features were removed due to its interference with the robot movement. Other cable management features were modified due to print defects. In some narrow parts while installing the thread inserts the soldering iron touched the part and caused it to melt.

6.6 Safety Considerations

Some safety considerations were taken during the assembly process. Appling forces on differnt part of the robot to make sure that the clearance between the parts is enough to avoid the parts touching each other during the robot movement.The clearance between the hip knee links and the wheel motor covers was 2 mm, the clearance between hip knee links and wheel knee links was 2 mm. Soldering iron was used to insert the thread inserts into the chassis, so the soldering iron was kept away from the chassis to avoid damaging it. Considering that long wires would trip the robot and short wires would not allow the robot to move freely and may damage the wires, the wires were cut to the required length to avoid these problems.

7 Software and Firmware Development

This chapter discusses the software and firmware development for the robot. Providing a through explanation of the software and firmware that form the backbone of the robot's functionality. it outlines the process of developing, integrating, and management of these systems, emphasizing the importance of each component and its role in the overall system. The chapter stands as a reference to the complexity and sophistication of involved in developing advanced robotic systems, where every layer of the software and firmware stack is carefully designed and implemented to ensure the proper operation of the robot.

7.1 Overview

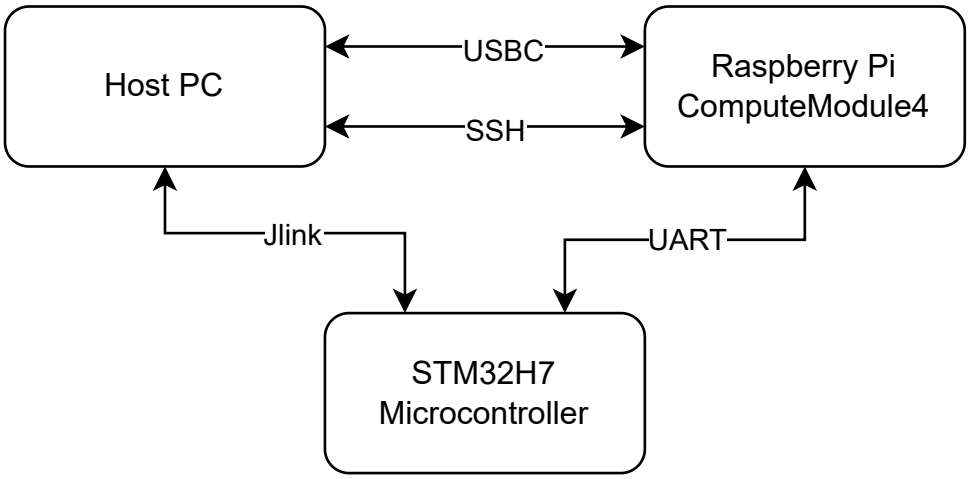


Figure 7.1: Hardware diagram

As shown in Figure7.1, system architecture is divided into three main parts: microcontroller firmware, raspberry pi software, and host pc software. The microcontroller firmware is responsible for controlling the robot's actuators and sensors. The raspberry pi software is responsible for high-level tasks of the robot. The host pc software is responsible for

7.2 Microcontroller Firmware

The microcontroller firmware is responsible for controlling the robot's actuators and sensors. As outlined in the electrical design Chapter, STM32H7 microcontroller is the main controller for the robot. The firmware is written in C language, with some parts written in C++. FreeRTOS is used as the real-time operating system for the microcontroller for task scheduling and management.

Feedback control is implemented in the firmware to control the robot's actuators. The feedback control is implemented using different control algorithms. Balancing the robot in different configurations is the main goal of the feedback control. State estimation is implemented in the firmware to estimate the robot's state and provide feedback to the control algorithms that generate the control signals for the actuators. Time critical functions are implemented in the firmware to ensure the real-time performance of the robot. These functions include the control loop for balancing the robot and some other functions that are time-critical such as safety protocols. The safety protocols are implemented to ensure the safety of the robot and the user. The safety protocols include the emergency stop for the motors in case of high current draw or high temperature. Such protocols ensure, for example, that the robot will not accelerate to a high speed and crash into an object.

The microcontroller firmware is responsible for different tasks that ensure the robot's proper operation. Tasks such as reading the sensors and filtering the sensor data are implemented in the firmware. The firmware is also responsible for controlling the motors and real time monitoring feedback from the motors to verify that the motors have reached the desired state as per the last control signal sent to the motors.

7.3 Raspberry Pi Software

The raspberry pi software is an important part of the robot's software. It is written in Python and deals with the high-level tasks of the robot. The raspberry pi allows wireless SSH connection to the robot, which allows the user to change the control parameters or control the robot using a joystick. The wireless connection allows different modifications to the robot's software without the need to connect the microcontroller to the computer as long as the raspberry pi and the pc are connected to the same network. The raspberry pi software is responsible for highr complex tasks such as speed control and position control. These tasks are not time-critical and can be implemented in the raspberry pi software. Then it can be interpreted by the microcontroller firmware into a series of low-level control signals for the actuators. Different applications such as navigation and mapping can be implemented in the raspberry pi software. Image processing and computer vision can be a powerful tool for the robot to interact with its environment.

The raspberry pi software allows conducting experiments and collecting data from the robot. The data acquired from the robot can be used to improve the robot's performance and control algorithms. The raspberry pi software is provides an easy to use python interface for controlling the robot.

7.4 Host PC Software

The host pc software is the third component of the robot's software. It is responsible for controlling multiple robots. It is also responsible for reading and writing parameters of individual or multiple robots. The changes in the parameters influence the behavior of the robot. The host pc software is also responsible for deploying firmware to multiple robots.

7.5 Integrated Development Environment

STM32CubeIDE is used as the IDE for firmware development. The STM32CubeIDE is based on the Eclipse IDE, which is an open-source IDE. The STM32CubeIDE allows clock configuration for the microcontroller. The STM32Cube also generates the Makefile for the project, which is used to compile the firmware.

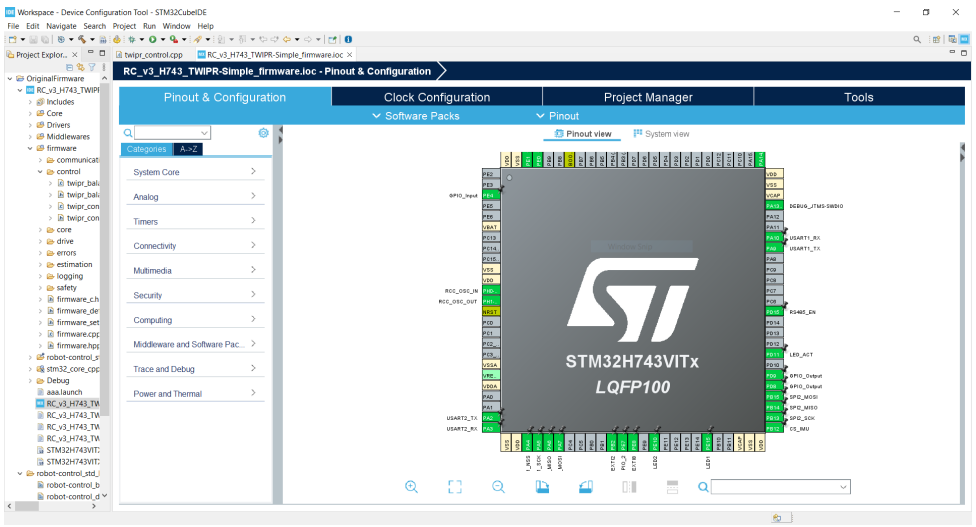


Figure 7.2: Pinout & Configuration

As shown in Figure 7.2, the STM32CubeIDE allows the user to configure the microcontroller peripherals, such as the GPIOs, timers, and UART. After configuring the microcontroller, the STM32CubeIDE generates the initialization code for the user.

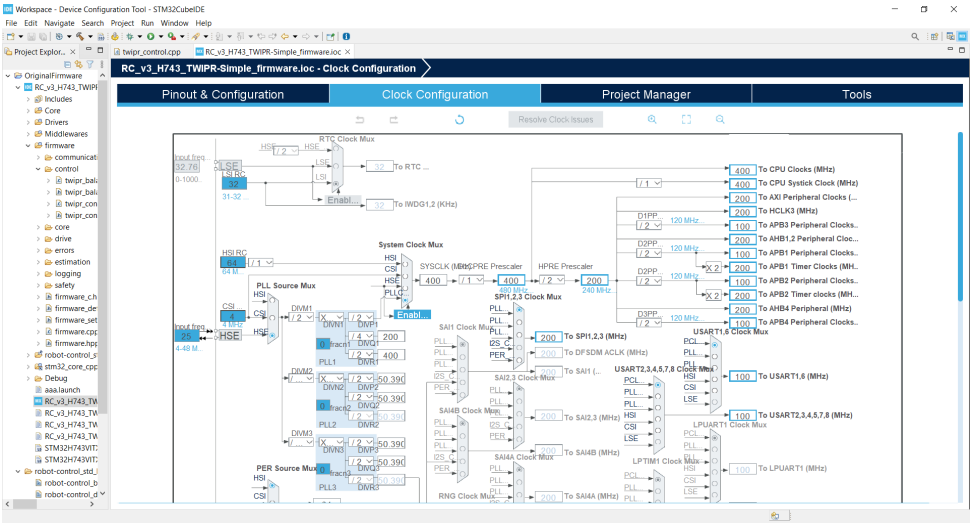


Figure 7.3: Clock Configuration

The STM32CubeIDE allows clock configuration for the microcontroller as shown in Figure 7.3.

7.6 Integration with Hardware

The robot hub Board is used to connect the microcontroller to the sensors and actuators. It also provides the power supply for the microcontroller as well as acting as a carrier board for raspberry pi. The robot hub board allows using the full potential of the microcontroller by providing the necessary connections.

7.7 Debugging and Troubleshooting

Figure 7.4: Jlink Debugger

Jlink s used to upload the firmware to the microcontroller. As well as debugging the firmware. The Jlink is a hardware debugger that connects to the microcontroller using the SWD interface. The Jlink allows debugging breakpoints and stepping through the code.

8 Conclusion and Future Work

8.1 Conclusion

In this research, we developed a multi-legged robotic system that can perform complex movements and interact with its environment. A new design was created from scratch to meet the requirements of the project. Throughout the design process, the mechanical, electrical, and software requirements of the robot were taken into account. Modeling of the new design was done to determine the robot's dynamics. The modeling includes the new robot's kinematics and dynamics integrating the changing location of the center of mass of the robot as well as the changing moments of inertia of the robot. the model was used to simulate the robot's motion and control. the simulation was used to test different control strategies and simulate the dynamic behavior of the robot.

The new design serves as a platform for future research and development. The new design is modular and can be easily modified to accommodate different requirements. The new design is also easily repairable. The simulation results prove the validity of the model and the effectiveness of the control strategies. The simulation results also show that the robot can change its configuration and maintain its balance.

Different challenges were faced during the development of the robot. The main challenges were the mechanical design and the control of the new model in different configurations. There are still some challenges and limitations that need to be addressed in future work.

8.2 Future Work

In future work, Refinement of the design for efficiency, stability. The next generation design can be more compact and lighter. Modeling as well can still be improved to include more details and more accurate parameters. Additionally, implementation of more advanced control strategies can be done. Different advanced capabilities can be integrated into the robot such as autonomous navigation, machine-learning-based control systems, or enhanced interaction with the environment. The new robot design can take advantage of the two independent legs to perform more complex movements such as bending one knee more than the other in fast and tight turns. Collaborative and swarm robotics can be explored, where multiple robots can work together to achieve a common goal.

Bibliography

[1] Boston Dynamics. Handle. https://www.bostondynamics.com/handle, 2021. Accessed: 2021-08-01.

[2] Victor Klemm, Alessandro Morra, Ciro Salzmann, Florian Tschopp, Karen Bodie, Lionel Gulich, Nicola Küng, Dominik Mannhart, Corentin Pfister, Marcus Vierneisel, et al. Ascento: A two-wheeled jumping robot. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7515–7521. IEEE, 2019.

[3] Honda. Asimo. https://asimo.honda.com/, 2021. Accessed: 2021-08-01.

[4] Boston Dynamics. Atlas. https://www.bostondynamics.com/atlas, 2021. Accessed: 2021-08-01.

[5] Sony. Qrio. https://www.sony.net/SonyInfo/technology/technology/theme/qrio/, 2021. Accessed: 2021-08-01.

[6] Patrick Klokowski, Julian Eßer, Nils Gramse, Benedikt Pschera, Marc Plitt, Frido Feldmeier, Shubham Bajpai, Christian Jestel, Nicolas Bach, Oliver Urbann, et al. evobot–design and learning-based control of a two-wheeled compound inverted pendulum robot. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10425–10432. IEEE, 2023.

[7] David Stoll. Development of a robot-in-the-loop testbed for cooperative learning in multi-agent systems, 2021.

[8] Alok Nath, Goutam Das, Anik Mallick, and Shovan Chowdhury. Design, implementation and stabilization of a bipedal robot. In *AIP Conference Proceedings*, volume 1919. AIP Publishing, 2017.

[9] Vishnu Vardhan Madadi and Sabri Tosunoglu. Design and development of a biped robot. In *2007 International Symposium on Computational Intelligence in Robotics and Automation*, pages 243–247. IEEE, 2007.

[10] Tiezheng Guo, Jinhui Liu, Haonan Liang, Yitong Zhang, Wei Chen, Ximing Xia, Meiqing Wang, and Zhiming Wang. Design and dynamic analysis of jumping wheel-legged robot in complex terrain environment. *Frontiers in Neurorobotics*, 16:1066714, 2022.

[11] Chao Zhang, Tangyou Liu, Shuang Song, Jiaole Wang, and Max Q-H Meng. Dynamic wheeled motion control of wheel-biped transformable robots. *Biomimetic Intelligence and Robotics*, 2(2): 100027, 2022.

[12] Zemin Cui, Yaxian Xin, Shuyun Liu, Xuewen Rong, and Yibin Li. Modeling and control of a wheeled biped robot. *Micromachines*, 13(5):747, 2022.

[13] Chun-Fei Hsu, Bo-Rui Chen, and Zi-Ling Lin. Implementation and control of a wheeled bipedal robot using a fuzzy logic approach. In *Actuators*, volume 11, page 357. MDPI, 2022.

[14] Songyan Xin and Sethu Vijayakumar. Online dynamic motion planning and control for wheeled biped robots. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3892–3899. IEEE, 2020.

[15] Sangtae Kim and SangJoo Kwon. Dynamic modeling of a two-wheeled inverted pendulum balancing mobile robot. *International Journal of Control, Automation and Systems*, 13:926–933, 2015.

[16] Aminu Yahaya Zimit, Hwa Jen Yap, Mukhtar Fatihu Hamza, Indrazno Siradjuddin, Billy Hendrik, and Tutut Herawan. Modelling and experimental analysis two-wheeled self balance robot using pid controller. In *Computational Science and Its Applications–ICCSA 2018: 18th International Conference, Melbourne, VIC, Australia, July 2-5, 2018, Proceedings, Part II 18*, pages 683–698. Springer, 2018.

[17] Anthony M Bloch, Naomi Ehrich Leonard, and Jerrold E Marsden. Stabilization of the pendulum on a rotor arm by the method of controlled lagrangians. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 1, pages 500–505. IEEE, 1999.

[18] Mohammad Mahdi Azimi and Hamid Reza Koofigar. Model predictive control for a two wheeled self balancing robot. In *2013 First RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)*, pages 152–157. IEEE, 2013.

[19] Leilei Cui, Shuai Wang, Jingfan Zhang, Dongsheng Zhang, Jie Lai, Yu Zheng, Zhengyou Zhang, and Zhong-Ping Jiang. Learning-based balance control of wheel-legged robots. *IEEE Robotics and Automation Letters*, 6(4):7667–7674, 2021.

[20] Kang Xu, Shoukun Wang, Binkai Yue, Junzheng Wang, Hui Peng, Dongchen Liu, Zhihua Chen, and Mingxin Shi. Adaptive impedance control with variable target stiffness for wheel-legged robot on complex unknown terrain. *Mechatronics*, 69:102388, 2020.

[21] Ravi Raj and Andrzej Kos. A comprehensive study of mobile robot: history, developments, applications, and future research perspectives. *Applied Sciences*, 12(14):6951, 2022.

[22] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):712–733, 2020.

[23] Michael Meindl. Development and iterative learning control of a two-wheeled inverted pendulum robot. Masterarbeit, Hochschule Karlsruhe - Technik und Wirtschaft, 2019.

[24] S. Fuller, B. Greiner, J. Moore, R. Murray, R. van Paassen, and R. Yorke. The python control systems library (python-control). In *60th IEEE Conference on Decision and Control (CDC)*, pages 4875–4881. IEEE, 2021. URL https://python-control.org.