ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

Wheeled-Legged Balancing Robot

Team #3

GABRIEL GAO
(ngao4@illinois.edu)
JERRY WANG
(runxuan6@illinois.edu)
ZEHAO YUAN
(zehaoy2@illinois.edu)

TA: Tianxiang Zheng

December 6, 2023

Abstract

In modern urban environments, we constantly encounter the limitations of current robotic technology. Most robots are either wheeled – great for smooth surfaces but not for obstacles like stairs, or legged – which can handle rough terrain but tend to be slow and power-consuming. This presents a real challenge for using robots effectively in urban settings.

Our project aims to address this issue by developing a new type of robot, one that combines the best of both worlds. We propose a hybrid robot that uses both wheels and legs, designed to navigate quickly and efficiently in urban environments. At the end of this project, our hybrid wheeled-legged robot can perform various tasks, such as carrying heavy loads, moving across rough landscapes, and even jumping. A key feature of our design is a leg system that also acts as a dynamic suspension, allowing the robot to smoothly handle different types of urban terrains.

This paper describes the design process, verification, cost, and schedule of our project in detail. In the design section, we will discuss the design of four main sections of our robot: mechanical structure, electrical system, embedded software, and control system. In the verification section, we will discuss how we met our requirements. Finally, we will discuss the cost of the project and our overall schedule.

Contents

1 Introduction

The purpose of our project is to build a versatile robot that can traverse various landscapes efficiently in urban settings. The unique design of our wheeled-legged robot allows it to traverse smoothly through uneven landscapes and gives it the potential to go across stairs.

Our robot has satisfied all of its functionalities. In addition to transitioning from a static position to a balanced position without toppling, the robot can reach the speed of 1.5 m/s. In addition. the robot can carry about 3.5 kg (8 lbs) of load, which is around its weight. We have also demonstrated that the robot can jump at a height of 15 centimeters (from wheel to ground).

As shown in figure 1, our system contains the power unit, the controller unit, and the actuator unit. The control unit is the central hub of our system, where the microcontroller sends and receives signals from different devices. The power unit is responsible for converting the battery voltage to 5V and 3.3V and supplying them to other parts of the system. The actuator unit consists of four leg motors and two wheel motors, which are all supplied by 24V from the power unit.

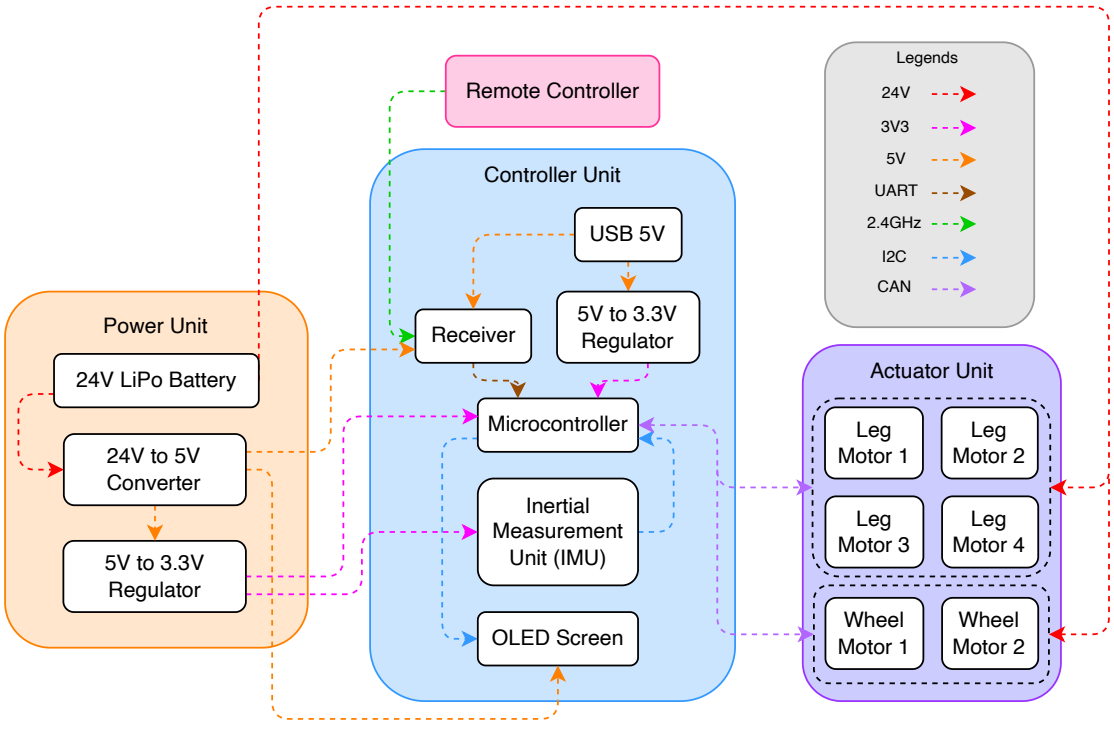

Figure 1: Top-level Block Diagram

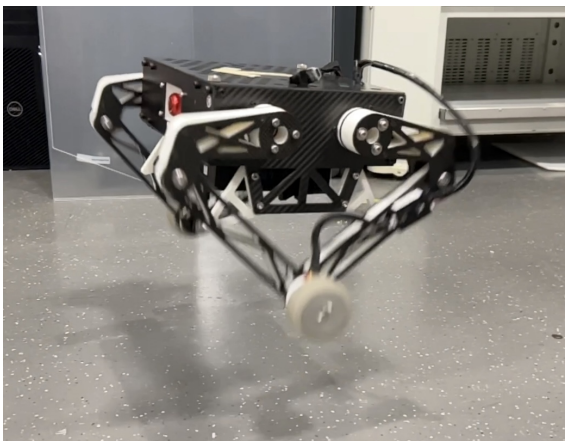

Figure 2: Robot Jumping

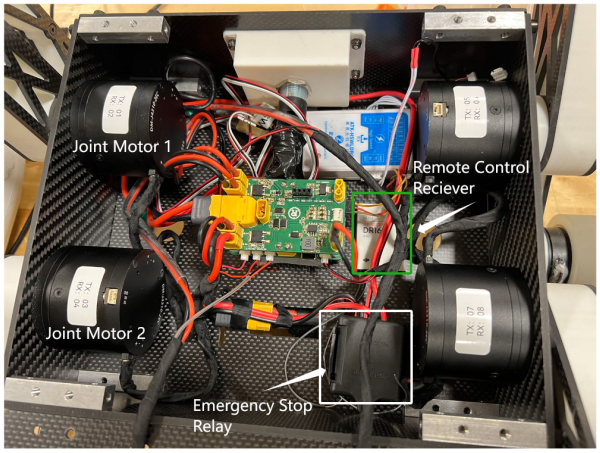

Figure 3: Wiring Inside the Robot

2 Design

2.1 Design procedure

2.1.1 Mechanical Structure

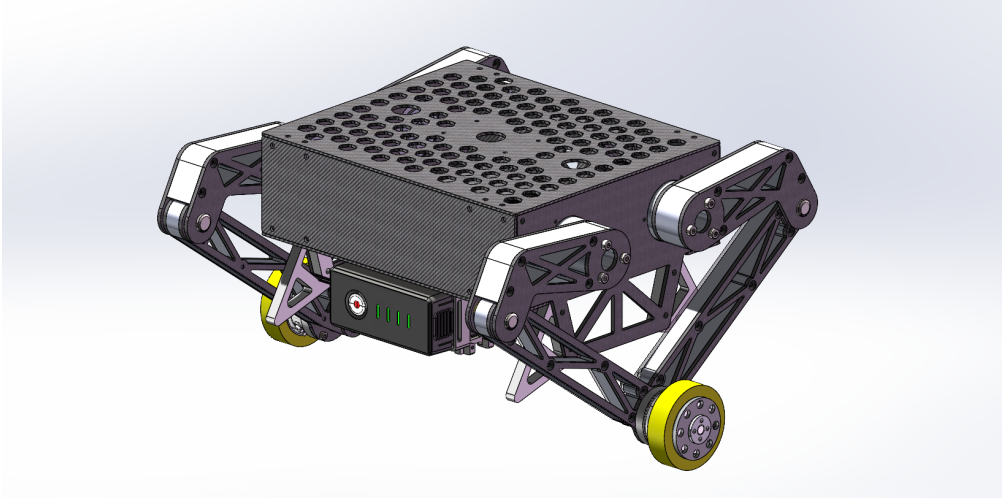

Figure 4: Assembly View of Robot

In the development of the mechanical structure for our hybrid wheel-legged robot, our approach was driven by the need for a high-performance, durable, and balanced design. The primary focus was on the robotic legs, conceptualized as an electronic suspension system. This required the legs to have motors with both high torque and rapid response capabilities to ensure the robot could effectively handle dynamic movements like jumping and navigating uneven terrain.

We carefully considered the strength and design of the legs to endure impacts and provide

stability, along with selecting wheels that offered the right size and friction for stable movement and speed.

Attention was also given to the robot's overall design, particularly its center of gravity. A symmetrical shape was chosen to distribute weight evenly, a key factor for stability and fluid motion. This phase included a meticulous selection of materials aimed at achieving a robust structure without compromising on the robot's lightness and agility.

2.1.2 Electrical System

The electrical system contains two separate PCB boards: a power board and a microcontroller board. The function of the power board is to convert the battery voltage from 24V to 5V and 3.3V, while the function of the micro-controller board is to communicate with each device and run the main program.

Our rationale for separating our circuit into two boards is that we can test them separately. If the power board fails during testing, it will not affect the microcontroller board. The two XT30 connectors allow the power board to supply power to the micro-controller board, and also act as pillars for the two boards to be stacked together.

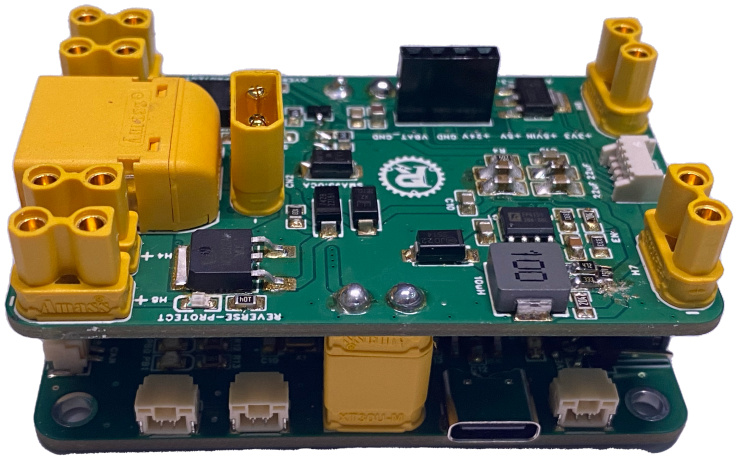

Figure 5: electrical system with DC-DC on top and Micro Controller on bottom

Power Board

In the design procedure of our DC-DC Power Board, we focused on creating a versatile and reliable power management system for connecting all motors and supplying two kinds of voltage levels to the microcontroller. The board is designed to handle a wide range of input voltages, specifically from 10 to 36 volts (battery at 24V), while providing stable and efficient output voltages crucial for the operation of various electric components like remote control receiver, UART port, etc. Safety features integral to the design: reverse polarity protection to prevent damage from incorrect input connections, and overvoltage protection to shield sensitive micro-controller boards from potentially harmful

high voltages. We also prioritized user-friendly monitoring and status indication, incorporating test points for voltage level checks and LED indicators for immediate visual feedback on the board's status.

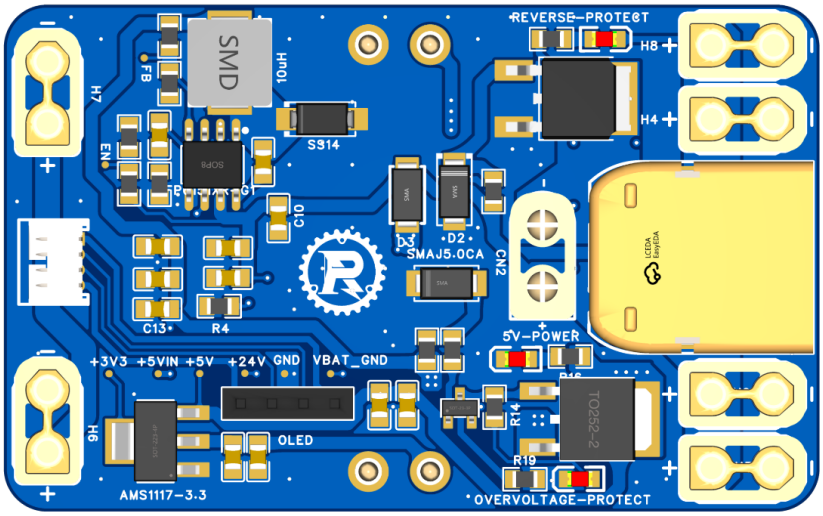

Figure 6: DC-DC Power Board

Microcontroller Board

For the microcontroller board, we are using STM32F103C8T6, a low-power ARM Cortex-M3 core MCU that supports a wide range of peripherals. However, the performance of this MCU is not especially high. The chip only has a 64KB flash memory and does not have dedicated DSP instructions, which means it is less efficient in performing DSP operations. At one point we even encountered FLASH overflow. Fortunately, this was solved using compiler optimization. For our future design, we can change to an ARM Cortex-M4 core MCU such as the STM32F407 series for higher performance.

Our board also contains an AMS1117 voltage regulator, a MAX3051 CAN Transceiver, and a BMI088 IMU. In addition, it includes a range of connectors: 2 XT30 connectors and 1 USB-C connector for power input, 5 ports for CAN, a port for DBUS, a port for Serial Wire Debug (SWD), and a port for UART.

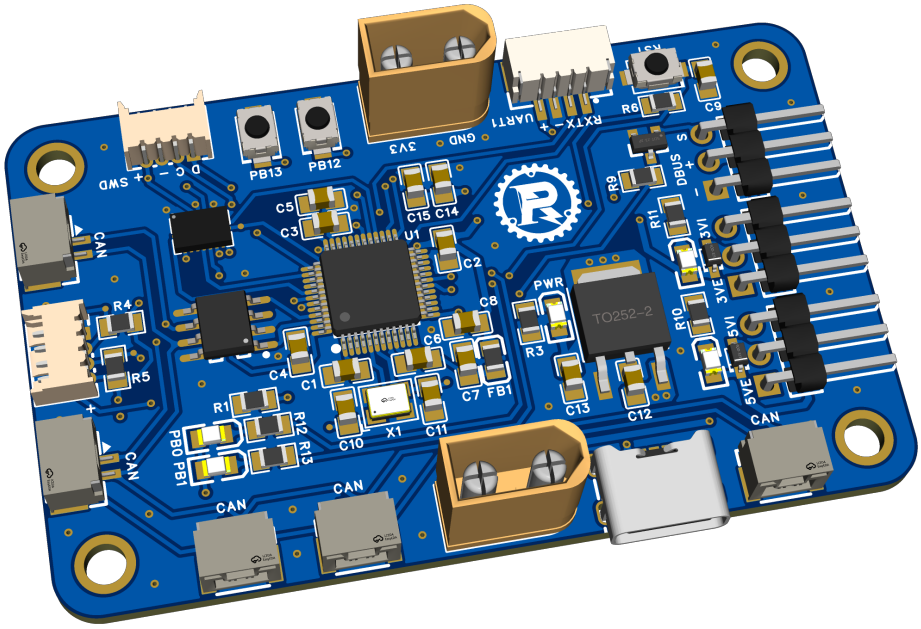

Figure 7: Render of the Microcontroller Board

2.1.3 Embedded Software

To communicate with our peripherals, we are using the API from the Hardware Abstraction Layer (HAL) provided by the chip manufacturer STMicroelectronics. We are also using a Real-time Operating System (RTOS) to split our program into multiple threads. Apart from multitasking, another reason for using RTOS is that it provides deterministic response times to events, which is important for us to process IMU data.

2.1.4 Control System

For the control system, we first establish the physical models of wheel motion and leg motion by using classical mechanical analysis. For the wheel motion model, there are three states: moving forward and backward, self-balancing, and rotation. For the leg motion model, we need to analyze the support state and the five-link model for each leg. Then, we introduce the linear quadratic regulator(LQR) algorithm for the wheel motion model and the spring-damping system of virtual model control(VMC) algorithm for the leg motion model. The reason for choosing the spring-damping system is that we want the robot can have an adaptive suspension system to reduce the vibration of the robot body when moving on rough surfaces. After that, we can use the Simulink tool at Matlab to do the simulation test. we conduct velocity and rotation tests for wheel motion, and we conduct elevation and adaptive suspension tests for leg motion. After passing the simulation test, we will implement the control system in C++ and flash into the robot.

2.2 Design details

2.2.1 Mechanical Structure

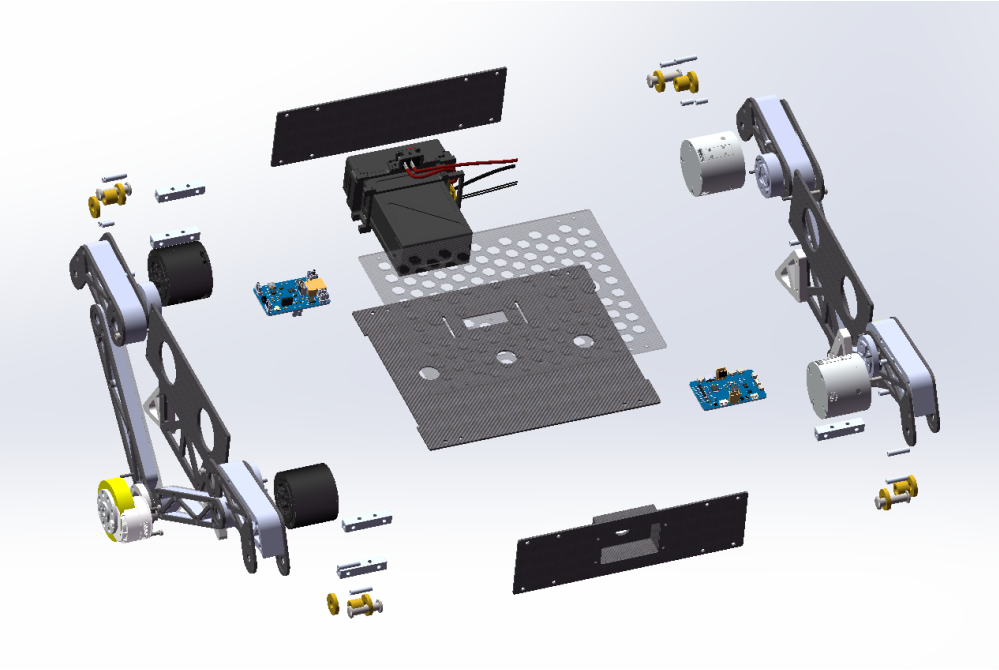

Figure 8: Exploded View

In the detailed design of our robot's mechanical structure, each component was chosen for its specific role in enhancing the robot's functionality. The selection of motors was a critical aspect. We opted for motors that not only provided high torque but also had rapid response capabilities. This choice was vital for the legs, enabling them to act effectively as an electronic suspension system. These motors are essential for activities like jumping and handling uneven ground, ensuring that the robot can cope with the various challenges of urban terrain.

The design also carefully considered the robot's center of gravity. A symmetrical shape was not only aesthetically pleasing but also functionally important for weight distribution. This symmetry was crucial in ensuring the robot's stability and smooth movement, particularly when navigating the varied and often unpredictable urban landscape.

We conducted static Force analysis on key components to ensure that they are as lightweight as possible while providing sufficient strength.

Figure 9: Emergency Stop mechanisms

Safety is a major concern in our robot project, given the inherent risks associated with its operation. To address this, we have incorporated three distinct emergency stop mechanisms, each designed to enhance safety for both the user and the robot during testing. Firstly, a mechanical emergency stop switch is located on the robot's body, allowing for immediate manual intervention. Secondly, a wireless emergency stop switch has been integrated into the remote control, providing a quick and effective way to halt the robot's operations from a distance. Lastly, for an additional layer of safety, we have implemented a program-level Kill Switch, also accessible via the remote control. These systems work independently, ensuring a comprehensive safety net during the robot's operation.

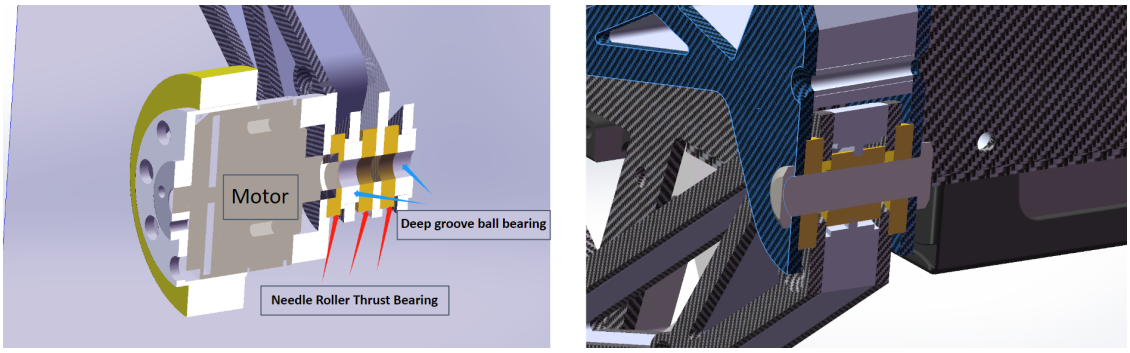

Figure 10: Joint Structure

Similar improvements have also appeared in the joints, where we have changed from thin PTFE sheets to a low-friction shaft system with multiple bearing combinations. Significantly reduce the frictional force of joint movement.

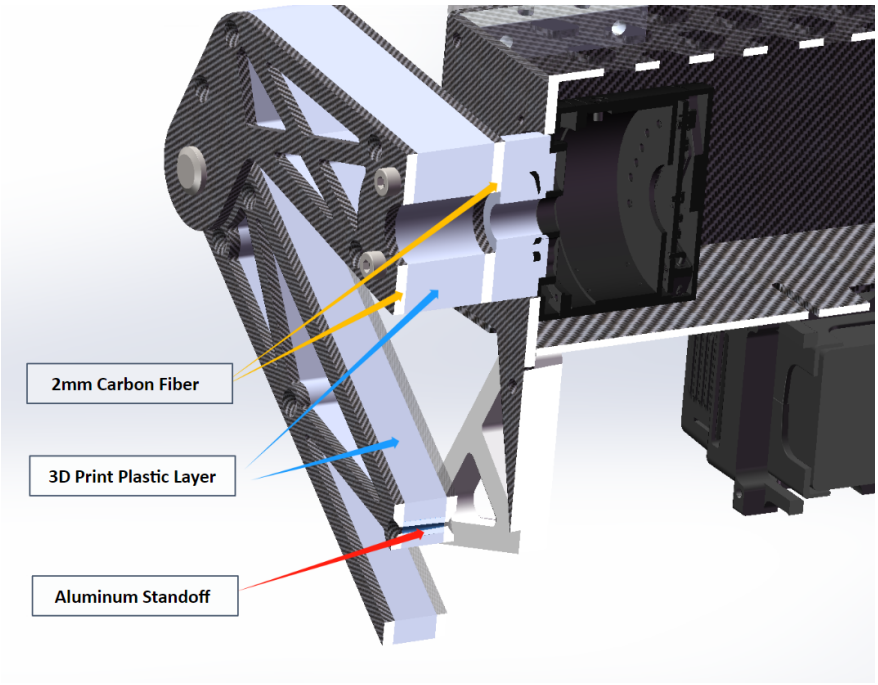

Figure 11: Leg 'sandwich structure' Cross Section

In the iterative design process of our robot, we evolved from an initial design featuring a single-layer thick aluminum plate to an enhanced version for greater durability. The original design, while lightweight, fell short in supporting lateral forces, particularly evident during side collisions on the leg components. To address this, we upgraded to a 'sandwich structure' design, comprising two layers of carbon fiber with an inner layer of low-density, 3D-printed material. This innovation significantly boosted the strength against lateral impacts while maintaining a similar weight to the original aluminum plate.

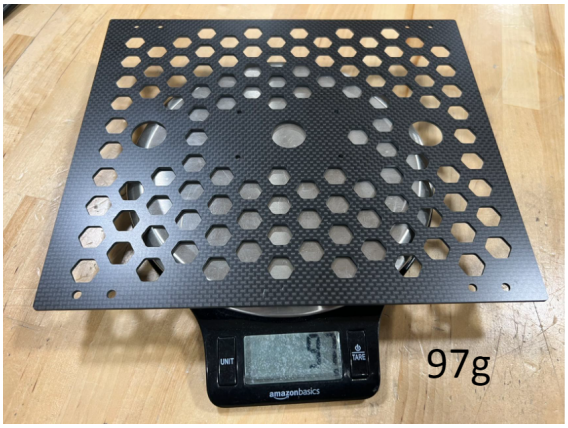

Figure 12: First Image Caption

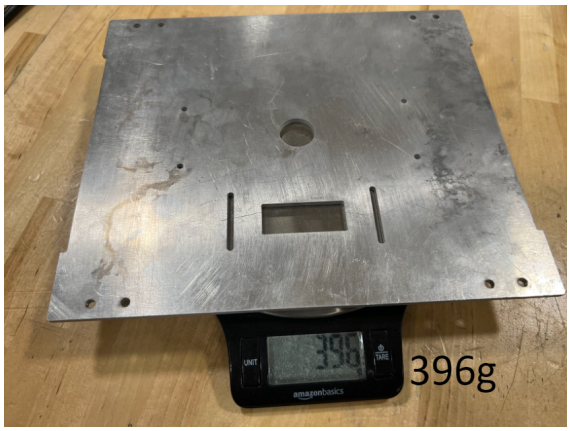

Figure 13: Second Image Caption

In our final material update, we significantly reduced the robot's weight by switching

from aluminum to carbon fiber as the primary structural material. Recognizing the superior strength-to-weight ratio of carbon fiber, this choice was a pivotal shift in our design strategy. In a comparative analysis, we found that the carbon fiber base of the robot was more than four times lighter than its aluminum counterpart. This weight reduction not only enhances the robot's load-carrying capacity but also contributes to improved overall performance. By opting for carbon fiber, we ensured that the robot remained robust yet agile, capable of higher functionality in various operational scenarios.

2.2.2 Electrical System

Power Board

The DC-DC Power Board's core comprises the FP6151 Buck Converter, chosen for its impressive efficiency, reaching up to 90%. This efficiency can minimize energy loss during voltage conversion. For the second Voltage level, we chose the AMS1117 Linear Voltage Regulator to ensure consistent voltage levels supplied to the microcontroller. The board's output capabilities include 5V up to 5A and 3.3V up to 1A, catering to a range of power requirements. The protection circuits are not only functional but also visually informative, with red LEDs indicating the activation of either the reverse polarity or overvoltage protection, and a green LED showing the circuit's operational status. Additional features enhance the board's functionality, such as the inclusion of six XT30 connectors for motor power distribution and an OLED screen (not shown in the picture) connector for displaying information from a microcontroller. These details collectively define a power board that is not only efficient and safe but also adaptable and user-friendly.

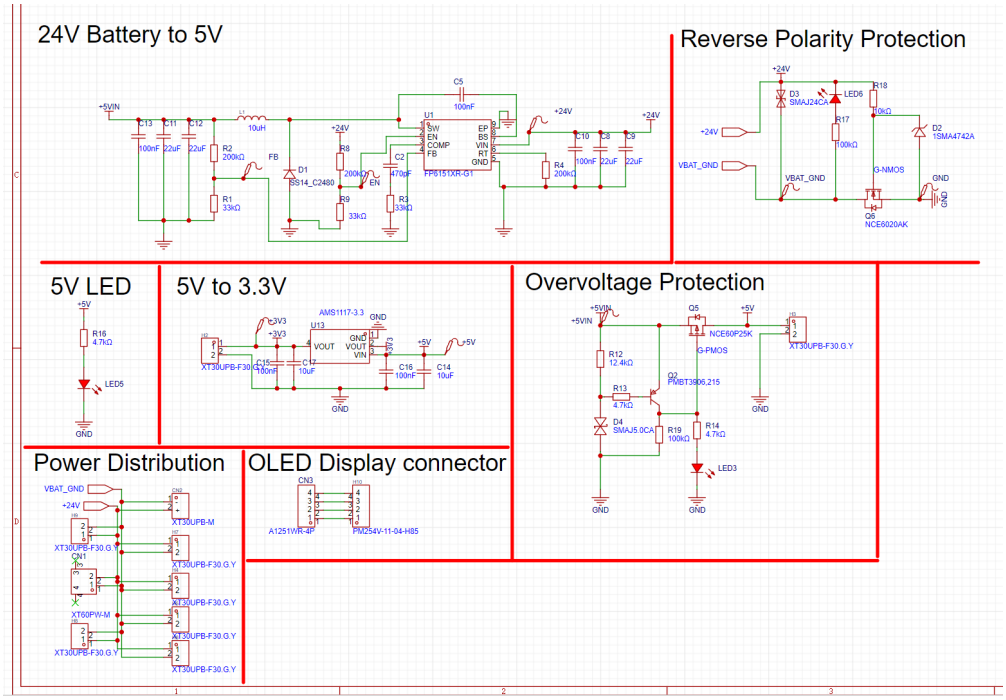

Figure 14: DC-DC Power Board Schematic

Microcontroller Board

The microcontroller board can be powered by two different power sources. When mounted on the robot, the board can be connected to the power board and receives 5V and 3.3V from it. If we want to test the board alone, we can power it through the USB-C port or the SWD port. In this case, the 5V supplied by our laptop will be converted to 3.3V by the onboard linear regulator. We can easily switch between these two types of power supply by switching the position of the jumper hats on the right side of the board. One of the jumper hats selects which 3.3V source it uses and the other selects which 5V source it uses.

We have also taken several design considerations when placing the PCB components. We placed the decoupling capacitors as close as possible to the MCU to minimize noise and stabilize the current supplied to the MCU. In addition, the IMU is placed at the farthest distance from the power circuit to minimize the effect of the heat generated by the linear regulator. This is because the IMU is a Micro-Electro-Mechanical System (MEMS), meaning that it contains tiny mechanical components, which are used to determine its angular velocity and linear acceleration. A changing temperature can cause these components to deflect, giving inaccurate sensor data.

The board also contains LEDs and push buttons, which can be configured for debugging purposes. In our main program, the LED is configured as a breathing light that indicates our program is running.

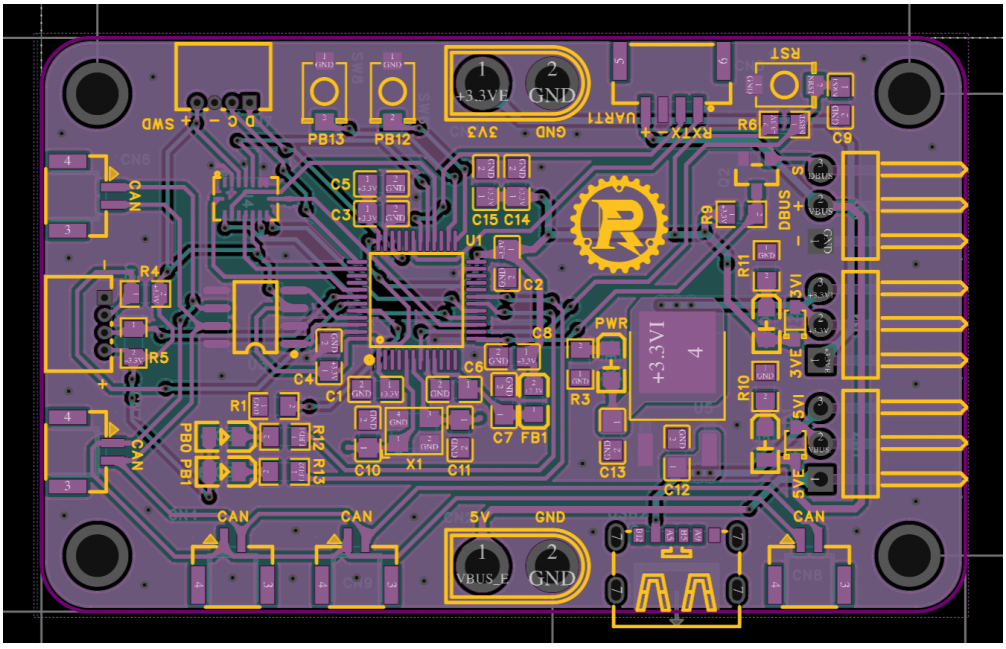

Figure 15: Layout of the Microcontroller Board

2.2.3 Embedded Software

Low-level Software

The HAL library that we are using provides functions that allow us to communicate with other devices. For example, functions like HAL_SPI_Transmit allow us to transmit messages to the IMU, and functions like HAL_CAN_RegisterCallback allow us to receive messages from the motors using hardware interrupts. We are also using Direct Memory Access (DMA) for some peripherals to reduce CPU load and speed up data transfer.

The RTOS allows us to create multiple tasks on a single core, where its scheduler is responsible for switching tasks, scheduling, etc. The RTOS kernel we are using is FreeRTOS, a popular open-source RTOS kernel. Above that, we are using the CMSIS-RTOS V2 library developed by ARM, which acts as an abstraction layer to FreeRTOS. Using this API, we can assign priority to each task, where the task with higher priority can preempt the ones with lower priority. For example, the IMU task requires the highest priority as it needs to process the raw IMU data in real-time.

Peripherals

The protocol we use to communicate with all of our motors is the Controller Area Network (CAN). An advantage of this protocol is that it allows all devices to communicate with each other without the need for a central hub. While we have six motors but only five CAN ports on our microcontroller board, we can connect two motors in series and connect one of them to the microcontroller.

For the IMU, we are using the Serial Peripheral Interface (SPI). The SPI is a high-speed full-duplex communication protocol, which is important for the IMU as we need real-time data for pose estimation.

We used the Universal Asynchronous Receiver/Transmitter (UART) protocol for two purposes: transmission from DBUS receiver and debugging. The remote control signals received by the receiver are transmitted to the microcontroller through a DBUS protocol, which is essentially an inverted UART signal. For debugging, we use UART to transmit messages to our screen through a serial port.

Sensor Fusion

Sensor fusion is crucial for precise orientation data. The IMU contains an accelerometer and a gyroscope, where the accelerometer measures linear acceleration and the gyroscope measures angular rate. Gyroscopes are prone to drift due to errors accumulated over time, while accelerometer measurements contain high-frequency noise. To get more reliable results, we can combine the high-frequency, short-term accuracy of the gyroscope with the low-frequency, long-term stability of the accelerometer.

The sensor fusion algorithm of our choice is the Mahony Filter [1]. Compared to algorithms such as the Extended Kalman filter (EKF), the Mahony Filter is less computationally intensive, which makes it more suitable for our MCU.

To use the Mahony Filter, we first obtain the sensor measurements from the IMU. We define ${}^{I}\omega_t$ as the gyroscope measurements and ${}^{I}\hat{\mathbf{a}}_t$ as the normalized accelerometer mea-

surements. Then, we calculate $\mathbf{v}\left({}^{I}_{W}\hat{\mathbf{q}}_{est,t}\right)$, the gravity component in the sensor's frame from the estimated quaternion at time t.

$$\mathbf{v}\left({}^{I}_{W}\hat{\mathbf{q}}_{est,t}\right) = \begin{bmatrix} 2\left(q_2 q_4 - q_1 q_3\right) \\ 2\left(q_1 q_2 + q_3 q_4\right) \\ \left(q_1^2 - q_2^2 - q_3^2 + q_4^2\right) \end{bmatrix} \tag{1}$$

Using $\mathbf{v}\left({}^{I}_{W}\hat{\mathbf{q}}_{est,t}\right)$ and the normalized accelerometer measurements ${}^{I}\hat{\mathbf{a}}_t$, we can use the cross product to calculate the error between these two vectors:

$$\mathbf{e}_{t+1} = {}^{I}\hat{\mathbf{a}}_{t+1} \times \mathbf{v}\left({}^{I}_{W}\hat{\mathbf{q}}_{est,t}\right) \tag{2}$$

Then, we can find the integral error as follows:

$$\mathbf{e}_{i,t+1} = \mathbf{e}_{i,t} + \mathbf{e}_{t+1}\Delta t \tag{3}$$

With the error $\mathbf{e}_{t+1}$ and its integral $\mathbf{e}_{i,t+1}$, we can perform the fusion step by updating the gyroscope measurements with a PI controller:

$${}^{I}\omega_{t+1} = {}^{I}\omega_{t+1} + \mathbf{K}_p\mathbf{e}_{t+1} + \mathbf{K}_i\mathbf{e}_{i,t+1} \tag{4}$$

Using the updated gyroscope measurements, we can calculate the change in orientation:

$${}^{I}_{W}\dot{\mathbf{q}}_{\omega,t+1} = \frac{1}{2}{}_{W}\hat{\mathbf{q}}_{est,t} \otimes \left[0, {}^{I}\omega_{t+1}\right]^T \tag{5}$$

where $\otimes$ represents quaternion multiplication.

Finally, we can update the orientation estimate with numerical integration:

$${}^{I}_{W}\mathbf{q}_{est,t+1} = {}^{I}_{W}\hat{\mathbf{q}}_{est,t} + {}^{I}_{W}\dot{\mathbf{q}}_{\omega,t+1}\Delta t \tag{6}$$

2.2.4 Control System

Wheel Motion Control System

For the analysis of wheel motion physical model, we have the following assumptions:

1. The mass of the body can be represented at the center of the mass.
2. Ignore the mass of the leg linkage and the effect on wheel motion from leg movement.
3. No sliding on the wheels.

The wheel motion can be separated into three parts: moving forward and backward, self-balancing, and rotation. The schematic of the wheel motion physical model is shown in **Figure 16**. The variable declaration is shown in **Appendix B** and the detailed mathematical proof is shown in **Appendix C** and **Appendix D**.

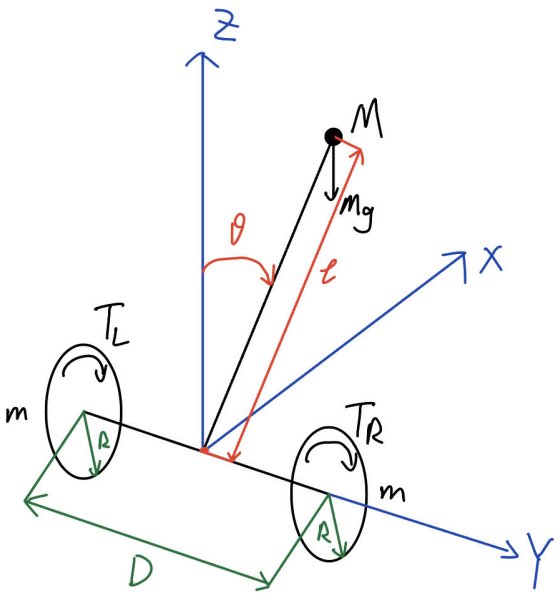

Figure 16: Simplified Wheel Motion Model

By analyzing the three situations of wheel motion and plugging in the parameter values, we can get the state space model $\dot{X} = AX + Bu$:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -14.7416 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 114.0117 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 21.0112 & 21.0112 \\ 0 & 0 \\ -105.5102 & -105.5102 \\ 0 & 0 \\ 67.6110 & -67.6110 \end{bmatrix} \cdot \begin{bmatrix} T_L \\ T_R \end{bmatrix} \tag{7}$$

To determine whether this system is controllable, we need to check whether the controllability matrix $C(A, B)$ is full rank.

$$C(A, B) = [B|AB|A^2B|...|A^5B], C(A, B) \in R^{6\times 12} \tag{8}$$

By using Matlab, $rank(C(A, B)) = 6$**, this system is controllable.**

Then we need to analyze the stability of the system. We can compute the eigenvalues of A to determine whether the current system is stable. If all eigenvalues are in the left Half Plane, the system is stable.
The eigenvalues of matrix A are:

$$\lambda = [0, 0, 10.6776, -10.6776, 0, 0]^T \tag{9}$$

This system is not stable, we need to design a controller to make this system stable. We will choose the LQR controller.
By using the LQR controller, we need to set the feedback gain matrix K:

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} \end{bmatrix} \tag{10}$$

Let $u = -KX$, we can rewrite the state space equation into $\hat{X} = AX - BKX = (A - BK)X$. We can let $A' = (A - BK)$ and change the value of each element in K to make the eigenvalues of A' all negative, which will make the whole system stable.

The cost function of the LQR controller is:

$$J = \int_0^\infty (X^T Q X + u^T R u) dt \tag{11}$$

We want to change the feedback controller $u = -KX$ to make the cost function become minimal, J_{min}. Matrix Q is a positive semi-definite matrix, which represents the punishment to the state vector X(or the error state vector). If the element in the Q matrix is larger, the corresponding element in the state vector will decrease to 0 more quickly. Matrix R is a positive definite matrix, which represents the punishment to the input vector. It is used to balance the importance of control inputs. Larger weights(elements in the R matrix) are usually assigned to control inputs when you want the system to consume less energy on those inputs or when you want them to be smoother.

According to the test, matrix Q and R used in this project are shown below:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 20000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 90000 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, R = \begin{bmatrix} 1500 & 0 \\ 0 & 1500 \end{bmatrix} \tag{12}$$

By using Matlab[2], the feedback gain matrix K is:

$$K = \begin{bmatrix} -0.0183 & -2.5897 & -9.2569 & -1.1094 & 2.5820 & 0.1954 \\ -0.0183 & -2.5897 & -9.2569 & -1.1094 & -2.5820 & -0.1954 \end{bmatrix} \tag{13}$$

Leg Motion Control System

For the analysis of leg motion physical model, we have the following assumptions:

1. The mass of the robot is effectively concentrated at the center of mass.
2. The influence of wheeled motion on leg motion is neglected.
3. The mass of the leg links is ignored.
4. The robot maintains a balanced state throughout the leg motion process, i.e., the pitch angle remains zero.
5. The two joint motors on the same side respond synchronously with equal magnitudes of control torque but in opposite directions.

When we do the classical mechanical analysis, We can equivalently represent the leg of the robot as a rigid bar whose length varies with the joint motor angles, and establish a statics model to support this relationship. The schematic model is shown in **Figure** 17. The variable declaration is shown in **Appendix B** and the detailed mathematical proof is shown in **Appendix C** and **Appendix D**.

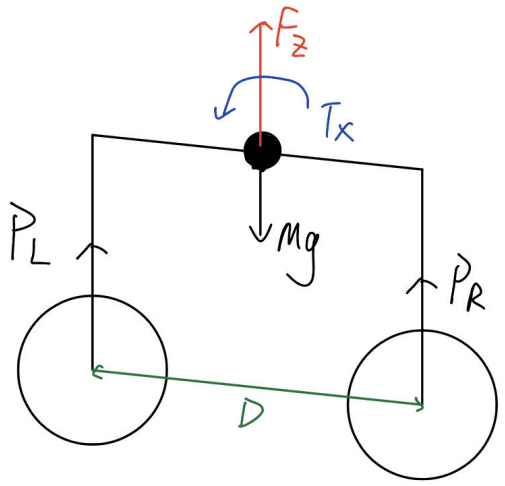

Figure 17: Simplified Support Phase Model

Based on this model, we can finally get the end-effector force equation:

$$\begin{cases} P_L = \dfrac{F_z + Mg}{2} - \dfrac{T_x}{D} \\ P_R = \dfrac{F_z + Mg}{2} + \dfrac{T_x}{D} \end{cases} \tag{14}$$

Then, each leg is a five-link model shown in **Figure** 18 and **Figure** 19. We can use the Pythagorean theorem to find the geometric relationship between the robot's height and its leg lengths as well as its leg motor angles.

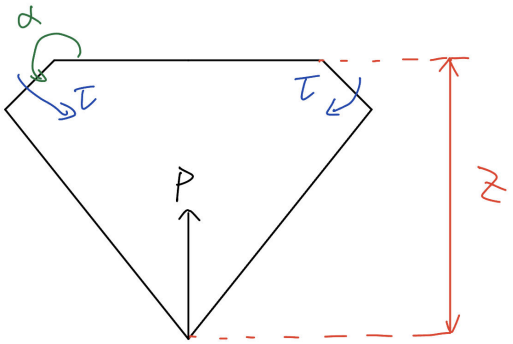

Figure 18: One side five-link model

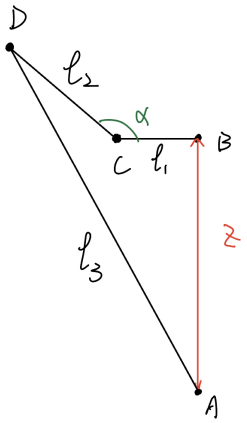

Figure 19: Half side five-link model

$$z = \sqrt{l_3^2 - (l_1 - l_2 cos(\alpha))^2} - l_2 sin(\alpha) \tag{15}$$

Furthermore, we can use the principle of virtual work to find the relationship between the leg motor torques and the end-effector forces.

$$\tau = -[\frac{(l_1 - l_2 cos(\alpha)) l_2 sin(\alpha)}{\sqrt{l_3^2 - (l_1 - l_2 cos(\alpha))^2}} + l_2 cos(\alpha)] * \frac{P}{2} \tag{16}$$

where P represents P_L or P_R in equation (14).

Because we want our robot legs to have adaptive suspension, we introduced **Virtual Model Control (VMC)** control algorithm. We chose the spring-damping system to represent our leg motion. The force and torque spring-damping system equations are shown in equation (17)

$$\begin{aligned} F_z &= k_1(z_{desired} - z) + c_1(0 - \dot{z}) = M\ddot{z} \\ T_x &= k_2\phi + c_2(0 - \dot{\phi}) = I_x\ddot{\phi} \end{aligned} \tag{17}$$

After plugging in equation (17) into equation (14), we can get the relationship below:

$$\begin{cases} P_L = \dfrac{k_1(z_{desired} - z) - c_1\dot{z} + Mg}{2} - \dfrac{k_2\phi - c_2\dot{\phi}}{D} \\ P_R = \dfrac{k_1(z_{desired} - z) - c_1\dot{z} + Mg}{2} + \dfrac{k_2\phi - c_2\dot{\phi}}{D} \end{cases} \tag{18}$$

$$\tau = -[\frac{(l_1 - l_2cos(\alpha))l_2sin(\alpha)}{\sqrt{l_3^2 - (l_1 - l_2cos(\alpha))^2}} + l_2cos(\alpha)] * \frac{P}{2} \quad \text{, where P is } P_L \text{ or } P_R \tag{19}$$

Combining motor rating data with simulation test performance, we will choose $k_1 = 400, c_1 = 70, k_2 = 120, c_2 = 10$.

3 Verification

All of our Requirements & Verification tables are in **Appendix** E.

3.1 DC-DC Power Board Test

To validate the functionality of our DC-DC power board, we conducted a series of tests using a DC power source and a battery. Initially, we set the power source at 20V and 27V and observed that the voltage output remained consistent at 5.06V and 3.30V, respectively, confirming that the board's voltage output aligns with our specifications.

For testing the output current, we connected the input to a battery and the output to various loads. Since we lacked a DC Electronic Load for a standardized test, we improvised by using motors as a practical load. The results showed that the output current on the 5V side exceeded 1A, satisfying our requirements. To assess the 3.3V output current, we connected it to a microcontroller board. The microcontroller board operated normally throughout the test, indicating that the current provided was adequate for our needs. These tests demonstrated that our DC-DC power board effectively meets the required voltage and current specifications for our project.

3.2 Microcontroller Test

The microcontroller is one of the most important parts of our system. It communicates with the IMU, the motors, and the radio receiver, meaning that its stable operation is crucial to our success. Because we are connecting 6 motors to the board through the same CAN bus, we need to ensure that the bus is not overloaded. To test this, we used a CAN analyzer that can monitor the load on a CAN bus. We tested the load on the bus during the robot's normal operation, which means all 6 motors are receiving commands through the same bus. The CAN analyzer showed the load on the CAN bus is around 40% on average during normal operation, far below the critical load threshold. Among the tests that we have performed on the robot, the CAN communication was always stable and we didn't notice any unusual behavior due to a broken communication.

3.3 Attitude Sensing Test

The attitude sensing of the robot relies on the IMU, which is mounted on our microcontroller board. To begin with, we have established stable communication with the IMU through SPI using hardware interrupt. The gyroscope and accelerometer data from the IMU is stable even under intense motion such as collision with the wall. One direct way to verify the attitude-sensing functionality is to observe the status of our robot, which heavily relies on the IMU to balance itself. We have shown that the robot can balance well using data from the IMU. We have also tested the drift of the IMU by placing the board in a fixed position for over 30 minutes. We compared the linear accelerations and angular velocities at the start and end of 30 minutes and observed that the difference in linear accelerations is less than 0.001 m/s and the difference in angular velocities is less than 0.003 m/s.

3.4 Wheel motion Simulation Test

The Simulink simulation block diagram is shown in **Appendix** F. For the wheel motion, we conduct the velocity test and rotation test. **For velocity test**, the robot takes 1 second to converge to our target velocity of 0.5 m/s without rotation. In this test, the maximum output torque of each wheel motor is 0.1437 Nm. **For rotation test**, the robot takes about 0.5 seconds to converge to our target rotation angle of 30 degrees almost without translation. In this test, the maximum torque of the left wheel is -0.2773 Nm and the maximum torque of the right wheel motor is 0.2773. **In both tests**, the maximum torque for each wheel motor is less than the theoretically maximum output torque of the motor.

Detailed test plots and data is shown in Appendix G

3.5 Leg motion Simulation Test

The Simulink simulation block diagram is shown in **Appendix** F. For the wheel motion, we conducted the elevation test and adaptive Suspension test. **For elevation test**, the robot takes 0.5 seconds to converge to our target elevation height of 0.2 meters. In the elevation test, the maximum output leg motor torque is 2.9 Nm. **For adaptive suspension test**, the robot tilted from an initial inclination of 0.2 rad to a horizontal position in 0.5 seconds. In this test, the maximum output torque of the left leg motor is 4.4 Nm and the maximum output torque of the left leg motor is -2.7 Nm. **In both tests**, the maximum torque for each leg motor is less than the theoretically maximum output torque of the motor.

Detailed test plots and data is shown in Appendix G

3.6 Remote Controller Test

To ensure the safety of the robot during operation, we have three levels of safety switches, including two on the remote controller. One of them is defined as the right switch on the remote controller, which acts as a software kill switch. This switch allows us to kill either the wheel motors only or all motors at once. During critical situations, we have proved that the switch can kill the motors in less than a second. The other switch is a remote relay switch, which can cut the power from the entire system remotely. This switch guarantees that if our program is jammed, we can still shut down the robot as fast as possible. We have tested that the switch can shut down the power of the robot completely in less than a second, which is crucial for the safety of the operator.

4 Cost and Schedule

4.1 Cost Analysis

The total cost without shipping cost in **Figure** 41(**Appendix** H) is $801.141. By adding 10% sales tax and 7% shipment cost, the total cost for parts is $937.34. For the labor cost, we expect we can have a salary of $\$40/hr * 2.5 * 65hr = \$6500$ per person. We have three team members. Therefore, the total labor cost should be $\$6500 * 3 = \$19500$. The machining costs at the machine shop are around $120. By adding the cost of the parts, the total cost of our project should be $\$19500 + \$937.34 + \$120 = \$20557.34$.

4.2 Schedule

Detailed Schedule for all team members is shown in **Appendix** H.

5 Conclusions

In this project, we designed the mechanical structure of the robot ourselves, employing a combination of carbon fiber plates and 3D-printed components to reduce weight and enhance strength. We achieved a separation of the power management module and the main control board. Successfully implementing multi-protocol communication and IMU sensor fusion, we also achieved a hybrid control of the wheels and leg motors. The robot are able to achieve our design goal. However, our current robot exhibits a slight high-frequency oscillation in the balanced state. This is attributed to fluctuations in the data transmitted from the IMU to the microcontroller, and there are also some disparities between our simulation model and the actual robot.

5.1 Future Work

To improve our design, To further optimize our design, we are considering various aspects. Regarding the **mechanical structure**, we plan to enhance the robot's stability by using larger wheels and a motor with higher output torque. From the **electrical system**

perspective, we intend to use a more powerful microcontroller and incorporate an IMU heating circuit on our PCB to reduce zero-drift issues in the IMU. For the **embedded software**, we can use more sophisticated IMU filtering algorithms such as Extended Kalman Filter (EKF) to mitigate the issue of IMU data fluctuations. Regarding the **control system**, we can remodel the leg linkage motion and add jump state analysis with ground detection.

5.2 Ethics Consideration

The structure of the wheeled-legged robot does not in itself present an ethical problem. However, when this structure is maliciously applied to some mobile robots, perhaps this will cause some ethical problems. Wheel-legged architectures can be built into automated mobile platforms that carry weapons or monitors. This would violate the **IEEE Code of Ethics**[3] and **Three Laws of Robotics**[4]. To avoid this situation, we will not be open-sourcing our core technical parts, such as PCB drawing, detailed motion modeling, and core code. At the same time, we will review our consumers(**avoiding malicious or illegal use**) and in the future will also apply our products in positive directions, such as urban logistics and transportation.

References

[1] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1217, 2008.

[2] Dawn Tilbury, Bill Messner, Rick Hill. "Control tutorials for matlab and simulink." (2021), [Online]. Available: https://ctms.engin.umich.edu/CTMS/index.php?aux=Home.

[3] IEEE. ""IEEE Code of Ethics"." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).

[4] Encyclopaedia, *Three Laws of Robotics*. Encyclopaedia Britannica, inc., 2023. [Online]. Available: https://www.britannica.com/topic/Three-Laws-of-Robotics.

[5] V. Klemm et al., "Ascento: A Two-Wheeled Jumping Robot," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7515–7521, 2019. DOI: 10.1109/ICRA.2019.8793792.

Appendix A Mechanical structure

A.1 Static Force Analysis

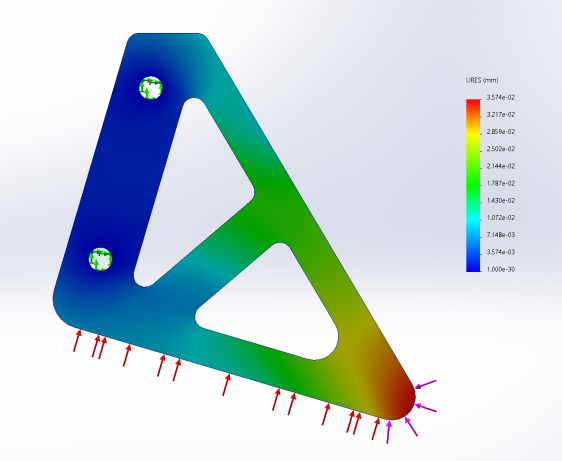

Figure 20: Static Force Analysis on Parking rack

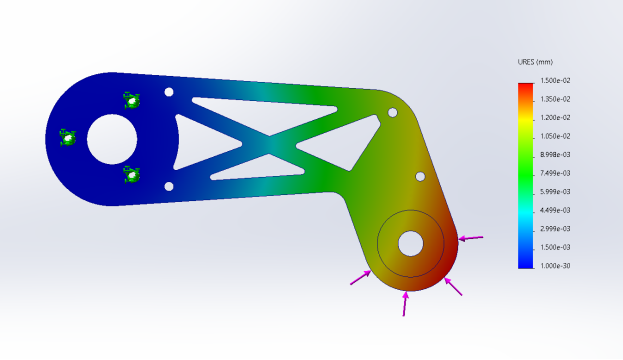

Figure 21: Static Force Analysis on Leg Component

Static force analysis shows Max URES on parts are about 3.5e-02(mm). Which would not cause Plastic deformation on the tested material.

A.2 Design Iteration

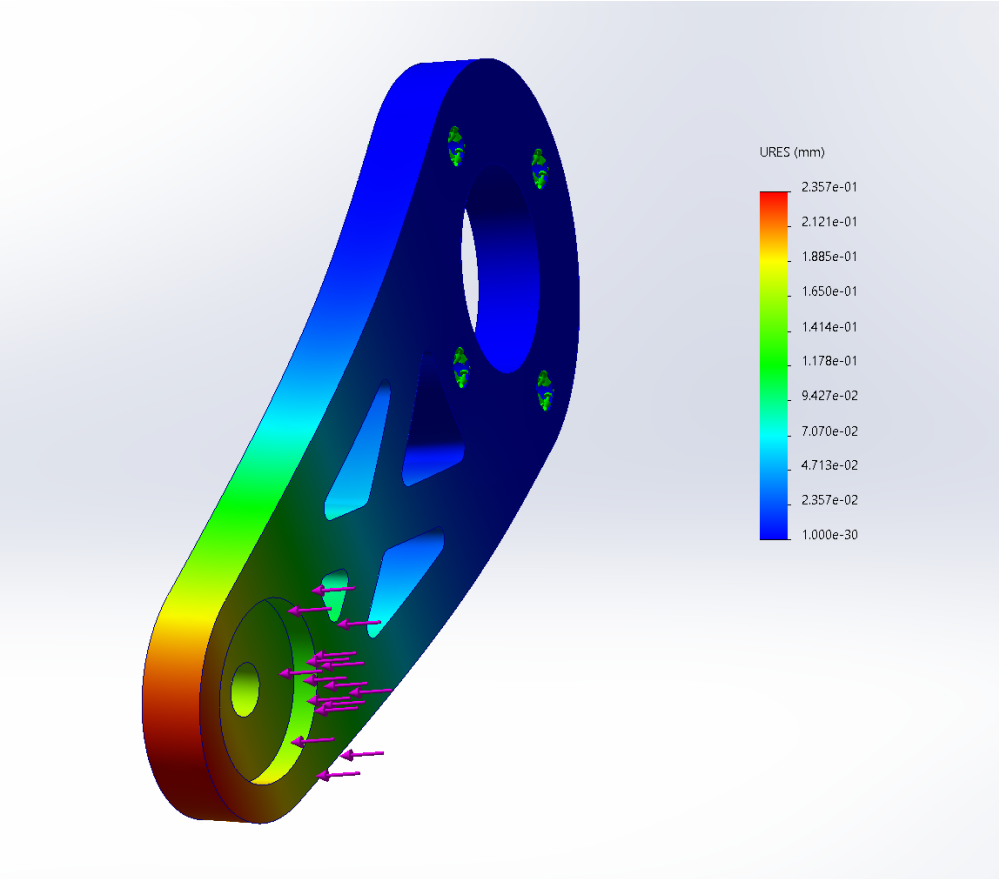

Figure 22: First Version

First version build by thick aluminum plate are more likely to bent. Simulation force applied is 100N. With Max URES 2.357e-01(mm).

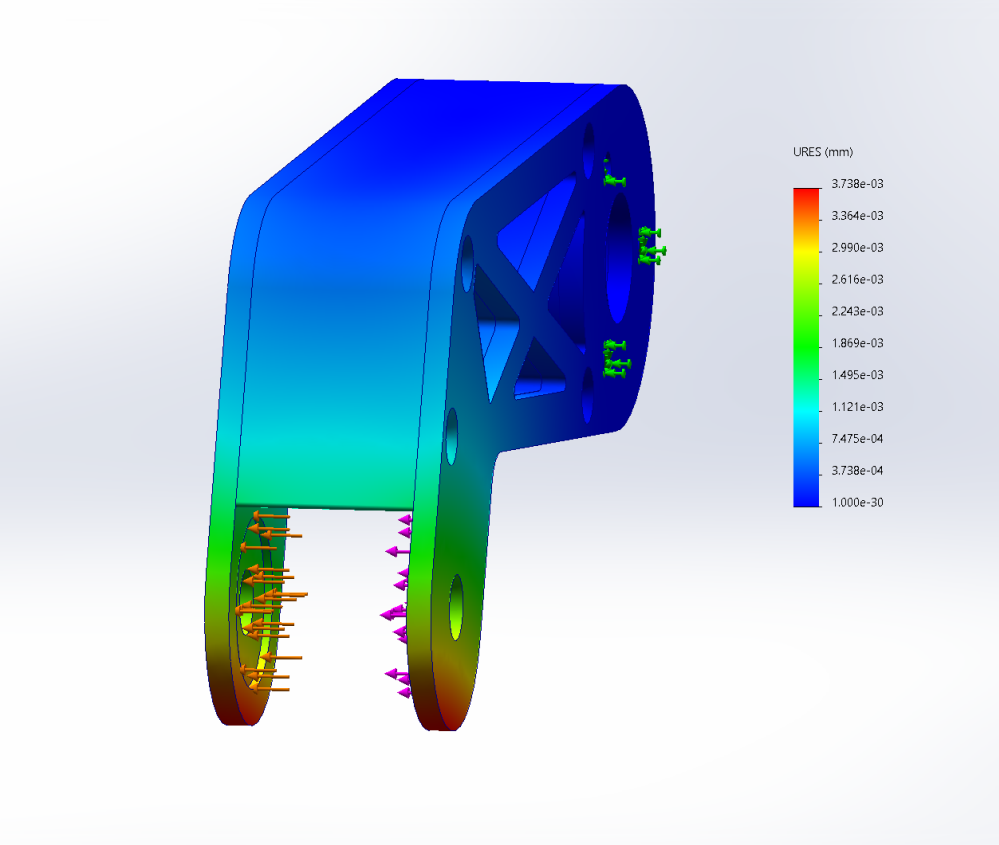

Figure 23: Enhanced Version

Enhanced version build by carbon fiber and plastic. Simulation force applied is 100N. With Max URES 3.738e-03(mm). Shown significant increase on the strength.

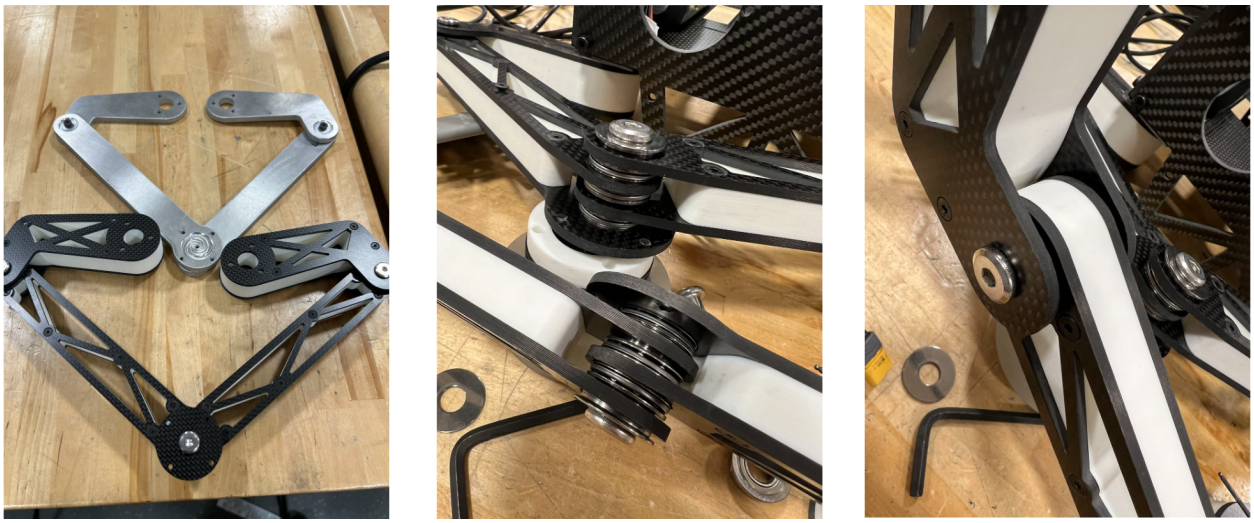

Figure 24: Comparison before and after optimization

Top left are the first version, build by thick aluminum plate, and the left bottom one are the final verion, with carbon fiber and 3D print plastic layer.The aluminum version weight 402 gram, carbon fiver version weight 373 gram. The middle figure shows the enhanced bearing structure. And the right figure shows the joint structure.

Appendix B Variables and Parameter Declaration Robot Physical Modeling

B.1 Wheel Motion Variable and Parameter Declaration and Values

Label	Meaning	Unit
x_L, x_R	The displacement of the left and right wheels.	m
z	The distance between the body's center of mass and wheel motor rotation axis along the z-axis.	m
ϕ	The roll angle of the body.	rad
θ	The pitch angle of the body.	rad
ψ	The yaw angle of the body.	rad
T_L, T_R	The output torque of the left and right wheel motors.	$N \cdot m$
T	The output torque of the leg motors.	$N \cdot m$
N_L, N_R	The horizontal component of the force between wheels and the body (along the x-axis).	N
P_L, P_R	The vertical component of the force between wheels and the body (along the y-axis).	N
F_L, F_R	The frictions of the wheels when moving.	N

Table 1: Variables

Label	Meaning	Unit
m	The mass of the rotor in the wheel motors.	kg
M	The mass of the body.	kg
I_w	The moment of inertia of the rotor in the wheel motors.	$kg \cdot m^2$
I_x	The moment of inertia of the body rotated around the x-axis.	$kg \cdot m^2$
I_y	The moment of inertia of the body rotated around the y-axis.	$kg \cdot m^2$
I_z	The moment of inertia of the body rotated around the z-axis.	$kg \cdot m^2$
R	The radius of the wheel.	m
l	The distance between the body's center of mass and the rotation axis of the wheel motor.	m
D	The distance between the left and right wheels.	m
g	The acceleration due to the gravity measured.	m/s^2

Table 2: Parameters

Label	Value	Unit
m	0.174	kg
M	3.848	kg
I_w	1.4741×10^{-4}	$kg \cdot m^2$
I_y	3.7049207×10^{-2}	$kg \cdot m^2$
I_z	6.563965×10^{-2}	$kg \cdot m^2$
R	0.03	m
l	0.152	m
D	0.3504	m
g	9.81	m/s^2

Table 3: Wheel Motion Parameters(with values)

B.2 Leg Motion Parameter Declaration and Values

Label	Meaning	Unit
l_1	Half of the length of the body's side.	m
l_2	The length of the active rod.	m
l_3	The length of the connecting rod.	m
z	The height of the body.	m
α	The angle of the leg motors.	$degree$

Table 4: Leg Motion Parameters

Label	Value	Unit
M	3.848	kg
I_x	7.0768703×10^{-2}	$kg \cdot m^2$
l_1	0.05	m
l_2	0.1	m
l_3	0.18	m
D	0.3504	m
g	9.81	m/s^2

Table 5: Leg Motion Parameters(with values)

Appendix C Analysis of Robot Physical Modeling

C.1 Wheel motion Physical Modeling [5]

C.1.1 Planar Motion: Moving Forward and Backward

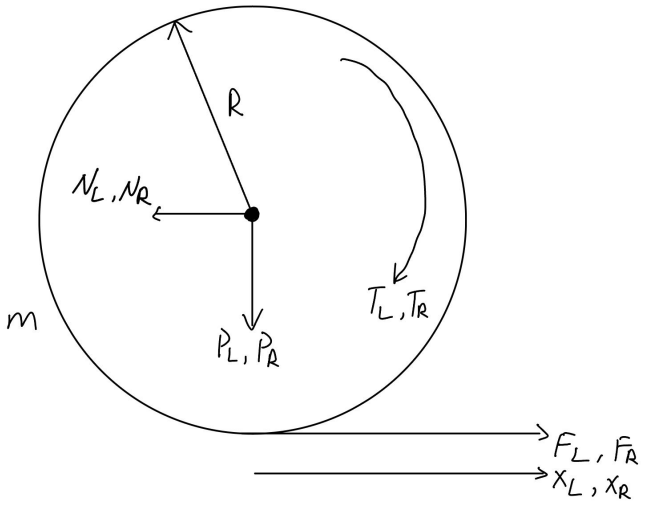

Figure 25: Wheel Model

For the net force:
Because we suppose the rotation for the left and right wheel motors is almost the same, we will use left wheel motor to do the calculation.

$$\begin{aligned} F_{net} &= ma \\ F_L - N_L &= m\ddot{x_L} \end{aligned} \tag{20}$$

For the net torque:
According to the transformation between rotation and linear motion:

$$\begin{aligned} \tau &= I\alpha \\ T_L - F_L * R &= I_w \frac{\ddot{x_L}}{R} \end{aligned} \tag{21}$$

Combine equation (20) and (21), we can eliminate F_L:
According (20):

$$F_L = m\ddot{x_L} + N_L \tag{22}$$

Plug (22) into (21):

$$\begin{aligned} T_L - (m\ddot{x_L} + N_L) * R &= I_w \frac{\ddot{x_L}}{R} \\ T_L - N_L R - mR\ddot{x_L} &= I_w \frac{\ddot{x_L}}{R} \\ (\frac{I_w}{R} + mR)\ddot{x_L} &= T_L - N_L R \\ \ddot{x_L} &= \frac{T_L - N_L R}{\frac{I_w}{R} + mR} \end{aligned} \tag{23}$$

we can also get the right wheel acceleration:

$$\ddot{x_R} = \frac{T_R - N_R R}{\frac{I_w}{R} + mR} \tag{24}$$

The acceleration of the whole robot is the average acceleration of the left and right wheels.

$$\begin{aligned} \ddot{x} &= \frac{\ddot{x_L} + \ddot{x_R}}{2} \\ \ddot{x} &= \frac{\frac{T_L - N_L R + T_R - N_R R}{\frac{I_w}{R} + mR}}{2} \\ \ddot{x} &= \frac{T_L + T_R - (N_L + N_R)R}{2(\frac{I_w}{R} + mR)} \end{aligned} \tag{25}$$

C.1.2 Planar Motion: Body Balance in Stationary State

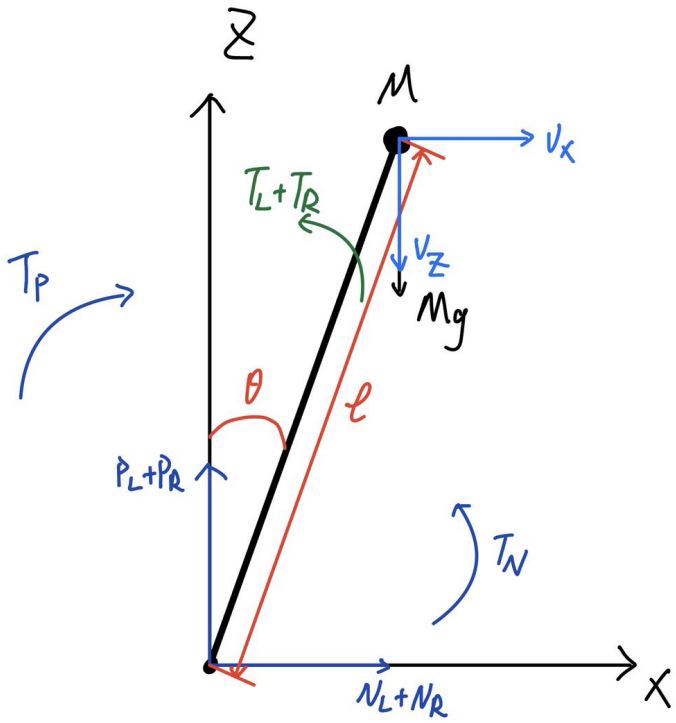

Figure 26: Self-Balancing Model

We suppose the force can be moved to the body's center of mass by convention.
We can decompose the velocity of the body into the horizontal and vertical directions.

$$v_x = \frac{\partial}{\partial t}(x + lsin(\theta)) = \dot{x} + l * cos(\theta)\dot{\theta} \tag{26}$$

$$v_z = \frac{\partial}{\partial t}(l - l * cos(\theta)) = l * sin(\theta) * \dot{\theta} \tag{27}$$

For the net force in Horizontal:

$$\begin{aligned} F_{net} &= ma \\ N_L + N_R &= M\dot{v_x} \\ N_L + N_R &= M(\ddot{x} + l * cos(\theta)\ddot{\theta} - l * sin(\theta)\dot{\theta}^2) \end{aligned} \tag{28}$$

For the net force in Vertical:

$$\begin{aligned} F_{net} &= ma \\ Mg - (P_L + P_R) &= M\dot{v_z} \\ P_L + P_R &= Mg - M(l * sin(\theta)\ddot{\theta} + l * cos(\theta)\dot{\theta}^2) \end{aligned} \tag{29}$$

By applying $N_L + N_R$ and $P_L + P_R$ to the body, we can get T_N and T_P.

$$T_N = (N_L + N_R) * l * cos(\theta), \quad T_P = (P_L + P_R) * l * sin(\theta) \tag{30}$$

For the net torque along the y-axis:

$$\begin{aligned} \tau &= I\alpha \\ I_y\ddot{\theta} &= T_P - T_N - (T_L + T_R) \end{aligned} \tag{31}$$

Combining the formulas (28), (29), (30), (31), we can eliminate $N_L, N_R, P_L, P_R, T_N, T_P$ to get the complete motion model of the body.

$$\begin{aligned} I_y\ddot{\theta} &= Mg * lsin(\theta) - M\ddot{x} * lcos(\theta) - Ml^2\ddot{\theta} - (T_L + T_R) \\ (I_y + Ml^2)\ddot{\theta} &= Mg * lsin(\theta) - M\ddot{x} * lcos(\theta) - (T_L + T_R) \end{aligned} \tag{32}$$

Combining the formulas (25) and (28), we can eliminate N_L, N_R to get the complete motion model of the wheels.

$$\begin{aligned} (\frac{2I_w}{R^2} + 2m)\ddot{x} &= \frac{T_L + T_R}{R} - M\ddot{x} - M * lcos(\theta)\ddot{\theta} + M * lsin(\theta)\dot{\theta}^2 \\ (\frac{2I_w}{R^2} + 2m + M)\ddot{x} &= \frac{T_L + T_R}{R} - M * lcos(\theta)\ddot{\theta} + M * lsin(\theta)\dot{\theta}^2 \end{aligned} \tag{33}$$

When the robot can keep the balance at a steady state by changing a tiny pitch angle(θ), we can linearize the parameters.

$$cos(\theta) = 1, \quad sin(\theta) = \theta, \quad \dot{\theta}^2 = 0 \tag{34}$$

By plugging (34) into (32) and (33), we can get the system of equations.

$$\begin{cases} (\frac{2I_w}{R^2} + 2m + M)\ddot{x} = \frac{T_L + T_R}{R} - Ml\ddot{\theta} \\ (I_y + Ml^2)\ddot{\theta} = Mgl\theta - M\ddot{x}l - (T_L + T_R) \end{cases} \tag{35}$$

C.1.3 Rotation Motion

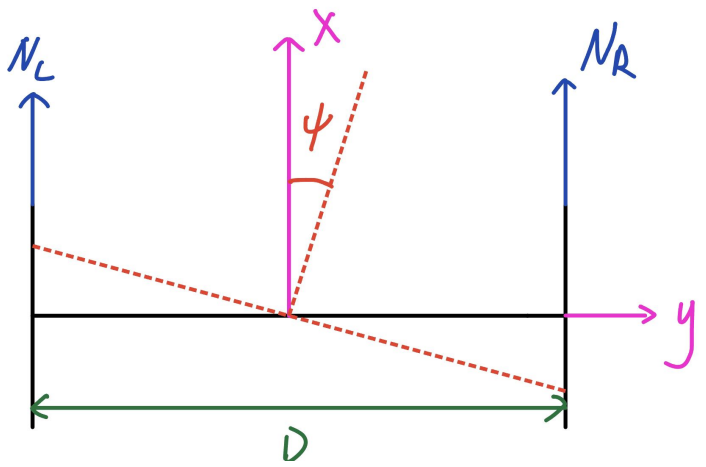

Figure 27: Rotation Model

For the net torque along z-axis:

$$\begin{aligned} \tau &= I\alpha \\ I_z\ddot{\psi} &= \frac{D}{2}(N_L - N_R) \end{aligned} \tag{36}$$

The relationship between the yaw angular acceleration and left and right wheel acceleration is:

$$\ddot{\psi} = \frac{\ddot{x_L} - \ddot{x_R}}{2} \tag{37}$$

Plugging equations (36) and (37) into equations (23) and (24), we can eliminate N_L and N_R :

$$\ddot{\psi} = \frac{T_L - T_R}{R(\frac{2I_z}{D} + \frac{I_w D}{R^2} + mD)} \tag{38}$$

C.2 Leg motion Physical Modeling

C.2.1 Five link Model

During the leg motion, the two leg motors of the robot have equal magnitudes of rotation angles but opposite directions, reducing the degrees of freedom of the five-bar linkage mechanism to one. Therefore, we can analyze only half of the five-bar linkage mechanism. The parameter declaration is shown in Appendix B.

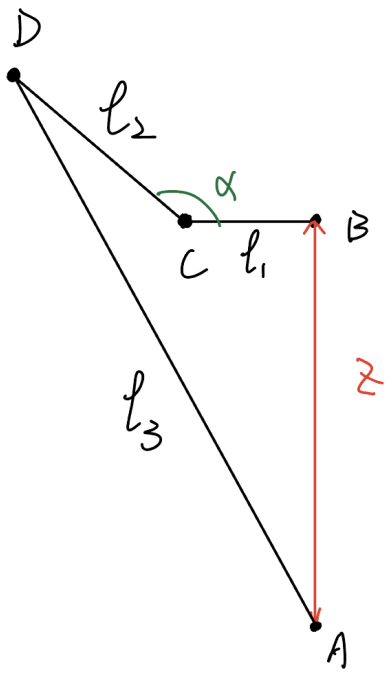

Figure 28: Simplified five-linkage Model

According to Figure 28 and the Pythagorean theorem, we can get the geometric relationship.

$$l_3^2 = (z + l_2 sin(\alpha))^2 + (l_1 - l_2 cos(\alpha))^2 \tag{39}$$

Because the length of each linkage is known, we can simply equation (39) to get the relationship between z and α.

$$z = \sqrt{l_3^2 - (l_1 - l_2 cos(\alpha))^2} - l_2 sin(\alpha) \tag{40}$$

C.2.2 Support Phase

When we do the classical mechanical analysis, We can equivalently represent the five-bar linkage of the leg as a rigid bar whose length varies with the joint motor angles, and establish a statics model to support this relationship.

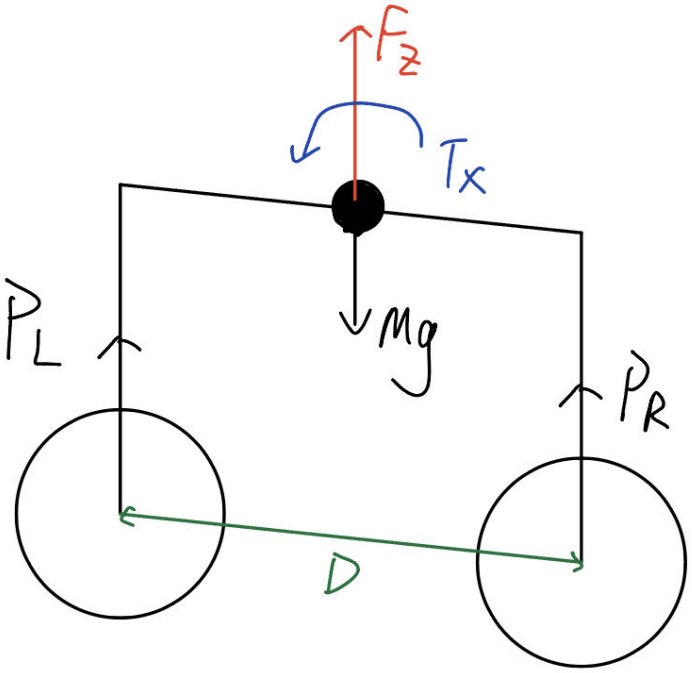

Figure 29: Simplified Support Phase Model

For the net force in the z-axis:

$$F_z = P_L + P_R - Mg \tag{41}$$

F_z is an abstract force that can control the movement along the z-axis to adjust the height of the robot.

$$F_z = M\ddot{z} \tag{42}$$

For the net torque along the x-axis

$$T_x = (P_R - P_L) * \frac{D}{2} \tag{43}$$

T_x is an abstract torque that can be used for controlling the robot's ground self-adaptive motion.

$$T_x = I_x \phi \tag{44}$$

By combining equation (41) and (43), we can get the following equations:

$$\begin{cases} P_L = \dfrac{F_z + Mg}{2} - \dfrac{T_x}{D} \\ P_R = \dfrac{F_z + Mg}{2} + \dfrac{T_x}{D} \end{cases} \tag{45}$$

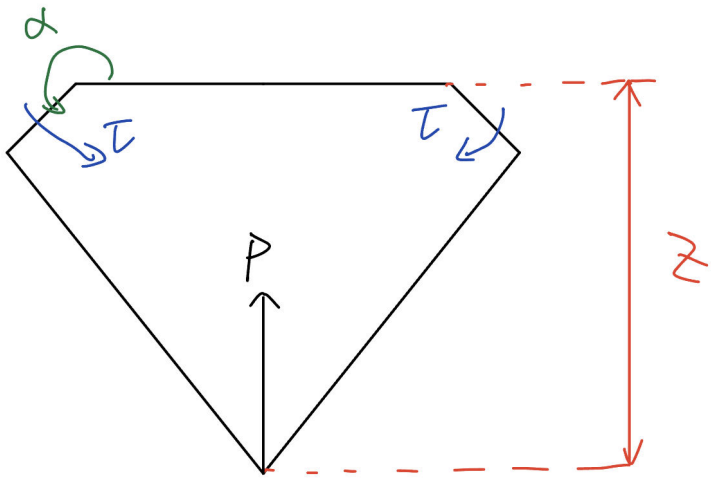

Figure 30: One side five-link model

Take one side leg as an example, we can establish the equations according to the principle of virtual work:

$$2\tau * \delta\alpha = P * \delta z \quad \text{P represents } P_L \text{ or } P_R. \tag{46}$$

By taking the derivative of equation (40), we can get:

$$\delta z = -[\frac{(l_1 - l_2 cos(\alpha)) l_2 sin(\alpha)}{\sqrt{l_3^2 - (l_1 - l_2 cos(\alpha))^2}} + l_2 cos(\alpha)]\delta\alpha \tag{47}$$

Plug in equation(47) to (46), we can get the output torque of the leg motors is:

$$\tau = -[\frac{(l_1 - l_2 cos(\alpha)) l_2 sin(\alpha)}{\sqrt{l_3^2 - (l_1 - l_2 cos(\alpha))^2}} + l_2 cos(\alpha)] * \frac{P}{2} \tag{48}$$

Appendix D Control Theories

D.1 Linear Quadratic Regulator(LQR) for Wheel Motion

According to equations (35) and (38), we can get the control equations:

$$\begin{cases} a\ddot{x} = T_L + T_R - b\ddot{\theta} \\ c\ddot{\theta} = d\theta - e\ddot{x} - (T_L + T_R) \\ \ddot{\psi} = f(T_L - T_R) \end{cases} \tag{49}$$

where

$$\begin{cases} a = R(\frac{2I_w}{R^2} + 2m + M) \\ b = MlR \\ c = I_y + Ml^2 \\ d = Mgl \\ e = Ml \\ f = \dfrac{1}{R(\frac{2I_z}{D} + \frac{I_w D}{R^2} + mD)} \end{cases} \tag{50}$$

The state space equation is

$$\begin{cases} \dot{X} = AX + Bu \\ Y = CX \end{cases} \tag{51}$$

We can determine the state vector X = $[x, \dot{x}, \theta, \dot{\theta}, \psi, \dot{\psi}]^T$ and the input vector u = $[T_L, T_R]^T$.

Then, we can convert equation(49) to the following form:

$$\begin{cases} \ddot{x} = \dfrac{-bd}{ac - be}\theta + \dfrac{c + b}{ac - be}(T_L + T_R) \\ \ddot{\theta} = \dfrac{ad}{ac - be}\theta - \dfrac{a + e}{ac - be}(T_L + T_R) \\ \ddot{\psi} = f(T_L - T_R) \end{cases} \tag{52}$$

Converting the above equation in the state space model:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-bd}{ac-be} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{ad}{ac-be} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{c+b}{ac-be} & \frac{c+b}{ac-be} \\ 0 & 0 \\ -\frac{a+e}{ac-be} & -\frac{a+e}{ac-be} \\ 0 & 0 \\ f & -f \end{bmatrix} \cdot \begin{bmatrix} T_L \\ T_R \end{bmatrix} \tag{53}$$

After plugging in the values of wheel motion parameters shown in **Appendix** B, we can get:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-bd}{ac-be} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{ad}{ac-be} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -14.7416 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 114.0117 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ \frac{c+b}{ac-be} & \frac{c+b}{ac-be} \\ 0 & 0 \\ -\frac{a+e}{ac-be} & -\frac{a+e}{ac-be} \\ 0 & 0 \\ f & -f \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 21.0112 & 21.0112 \\ 0 & 0 \\ -105.5102 & -105.5102 \\ 0 & 0 \\ 67.6110 & -67.6110 \end{bmatrix} \tag{54}$$

To determine whether this system is controllable, we need to check whether the controllability matrix $C(A, B)$ is full rank.

$$C(A, B) = [B|AB|A^2B|...|A^5B], C(A, B) \in R^{6\times 12} \tag{55}$$

By using Matlab, $rank(C(A, B)) = 6$**, this system is controllable.**

Then we need to analyze the stability of the system. We can compute the eigenvalues of A to determine whether the current system is stable. If all eigenvalues are in the LHP(left Half Plane), the system is stable.

The eigenvalues of matrix A are:

$$\lambda = [0, 0, 10.6776, -10.6776, 0, 0]^T \tag{56}$$

This system is not stable, we need to design a controller to make this system stable. We will choose the LQR controller.

By using the LQR controller, we need to set the feedback gain matrix K:

$$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} \end{bmatrix} \tag{57}$$

Let $u = -KX$, we can rewrite the state space equation into $\hat{X} = AX - BKX = (A - BK)X$. We can let $A' = (A - BK)$ and change the value of each element in K to make the eigenvalues of A' all negative, which will make the whole system stable.

The cost function is:

$$J = \int_0^\infty (X^T Q X + u^T R u) dt \tag{58}$$

We want to change the feedback controller $u = -KX$ to make the cost function become minimal, J_{min}. Matrix Q is a positive semi-definite matrix, which represents the punishment to the state vector X(or the error state vector). If the element in the Q matrix is larger, the corresponding element in the state vector will decrease to 0 more quickly. Matrix R is a positive definite matrix, which represents the punishment to the input vector. It is used to balance the importance of control inputs. Larger weights(elements in the R matrix) are usually assigned to control inputs when you want the system to consume less energy on those inputs or when you want them to be smoother.

The feedback gain matrix K is:

$$K = R^{-1} B^T P \tag{59}$$

Where matrix P is gotten from the algebraic Riccati equation.

In this project, we can use the Matlab LQR function to get the K directly.

According to the test, matrix Q and R used in this project are shown below:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 20000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 90000 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, R = \begin{bmatrix} 1500 & 0 \\ 0 & 1500 \end{bmatrix} \tag{60}$$

By using Matlab, the feedback gain matrix K is:

$$K = \begin{bmatrix} -0.0183 & -2.5897 & -9.2569 & -1.1094 & 2.5820 & 0.1954 \\ -0.0183 & -2.5897 & -9.2569 & -1.1094 & -2.5820 & -0.1954 \end{bmatrix} \tag{61}$$

D.2 Virtual Model Control(VMC) for Leg Motion

The leg motion control of the robot is implemented using the Virtual Model Control (VMC) method. This involves adding spring-damper virtual components to establish

the virtual forces required for leg motion. Subsequently, the end-effector forces at the leg links are determined using these virtual forces, and finally, the required torque for each joint motor is calculated based on the end-effector forces.

As shown in Figure 29, we have a virtual force along with z-axis, F_z and a virtual torque along with x-axis, T_x. We should establish the spring-damper virtual components, respectively.

For F_z, we can set the spring constant as k_1 and set the damping constant as c_1. Therefore, we can get the equation:

$$F_z = k_1(z_{desired} - z) + c_1(0 - \dot{z}) = M\ddot{z} \tag{62}$$

For T_x, we can set the spring constant as k_2 and set the damping constant as c_2. Therefore, we can get the equation:

$$T_x = k_2\phi + c_2(0 - \dot{\phi}) = I_x\ddot{\phi} \tag{63}$$

We can plug equation (62) and (63) into the equation (45), and we can get:

$$\begin{cases} P_L = \dfrac{k_1(z_{desired} - z) - c_1\dot{z} + Mg}{2} - \dfrac{k_2\phi - c_2\dot{\phi}}{D} \\ P_R = \dfrac{k_1(z_{desired} - z) - c_1\dot{z} + Mg}{2} + \dfrac{k_2\phi - c_2\dot{\phi}}{D} \end{cases} \tag{64}$$

After we have the forces(P_L, P_R) on the end-effectors of the leg, we can get the torque for each leg motor based on the equation (48).

$$\tau = -[\frac{(l_1 - l_2cos(\alpha))l_2sin(\alpha)}{\sqrt{l_3^2 - (l_1 - l_2cos(\alpha))^2}} + l_2cos(\alpha)] * \frac{P}{2} \quad \text{, where P is } P_L \text{ or } P_R \tag{65}$$

In the equation, α is a real-time value, which is gotten from the encoders in the motor. It is hard to introduce this value in the simulation. Therefore, we will use the average value of α during the simulation motion to implement the model.

Appendix E Requirement and Verification Table

E.1 Actuated Legs

Requirement	Verification	Result
• The motor should be able to supply a continuous torque of at least 4Nm ± 5% at 120RPM ± 5%.	• Using a script to read the torque value from the data given by the motors or Using the rotary torque sensor to measure the torque. Check whether the values from either method are larger than 4 Nm ± 5%. • Using a tachometer to measure the motor's RPM to confirm it can achieve 120 RPM ± 5% under the specified torque load.	Y
• The motors should be able to be powered by 20 - 27V.	• Using an adjustable power supply to test the motors at various voltages within the range of 20-27V to ensure consistent and efficient performance.	Y
• The motors must maintain a communication rate of 1kHz ± 10HZ with the STM32 through the CAN bus.	• Use a CAN bus analyzer or other appropriate testing tools to measure the communication rate between the motors and the STM32. Ensure the communication rate remains stable at 1kHz ± 10HZ.	Y
• The encoder of the motor should have an angle resolution of less than 1 degrees • Able to remember the correct motor zero point after power cutoff.	• Run multiple commands to rotate the motor to a certain angle, and measure whether each angle rotation matches the control command. Rotate the motor at a certain angle and read the motor angle feedback, comparing whether the angle feedback is within 1 degree of the actual difference. • After setting the motor to zero point, power off and rotate the motor angle. After powering on, check if the motor remembers the correct zero position	Y

Table 6: Actuated Legs–Requirements & Verification

E.2 Wheeled Drive

Requirement	Verification	Result
• M3508 motor maintains a communication rate of 1kHz ± 10Hz with the STM32.	• Use a CAN bus analyzer or other appropriate testing tools to measure the communication rate between the motors and the STM32. Confirm the communication rate remains stable at 1kHz ± 10HZ.	Y
• The motors should be able to be powered by 20 - 27V.	• Using an adjustable power supply to test the motors at various voltages within the range of 20-27V to ensure consistent and efficient performance.	Y
• M3508 motor delivers a continuous torque of at least 0.1Nm and achieves a speed of at least 2000 RPM. • M3508 motor can move the robot under a load of around 2.5lb.	• Using a script to read the torque value from the data given by the motors or Using the rotary torque sensor to measure the torque. Check whether the values from either method are larger than 0.1 Nm $\pm$ 5% • Using a tachometer to measure the motor's RPM to confirm it can achieve 2000 RPM $\pm$ 5% under the specified torque load. • Load the robot with a weight of 2.5lb and verify the motor moves the robot effectively.	Y
• The encoder of the motor should have an angle resolution of less than 1 degrees	• Run multiple commands to rotate the motor to a certain angle, and measure whether each angle rotation matches the control command. Rotate the motor at a certain angle and read the motor angle feedback, comparing whether the angle feedback is within 1 degree of the actual difference.	Y

Table 7: Wheeled Drive–Requirements & Verification

E.3 Attitude Sensing

Requirement	Verification	Result
• Low bias error and drift. • Low noise levels, High resolution	• Read the data from IMU in a normal environment and Compare it with the data from the reference sensor. • Let the PCB work at a high electromagnetic interference environment and check the return data with the reference sensor.	Y
• Suitable range and high sampling rate to get reasonable data.	• In a stable state of the IMU, connect the data output pins of the IMU to an oscilloscope and observe the waveform of the output signal. By measuring the time interval between two consecutive peaks, we can estimate the sampling rate.	Y
•Ability to be calibrated and temperature-compensated	• Use a script to send the calibrating signal to IMU and use the return data to check whether the IMU is calibrated. Then, Keep monitoring the temperature data from IMU during the operation to check whether it can be temperature-compensated.	Y
•Low power consumption	• Using the adjustable DC power supply, which can show the input voltage, input current, and input power, to measure the power consumption during operation.	Y
•Common interfaces (I2C, SPI, UART, or CAN)	• Using simple scrips to check whether the microcontroller can send the signals to IMU and receive the IMU signals.	Y

Table 8: Attitude Sensing – Requirements & Verification

E.4 PCB and Microcontroller

Requirement	Verification	Result
• Capable of processing inputs efficiently and directing outputs to various peripherals.	• Perform an on-off test and voltage drop test on the entire PCB board using a multimeter. After the first two tests are successful, power up the PCB board and measure the voltage of the circuit components to see if they are regular. Then using scripts to test the functionality of each interface.	Y
• Must have protection mechanisms against power surges or short circuits.	• Check the input and output capacitors before powering the PCB board to ensure they can filter voltage fluctuations. • Using the adjustable DC power supply to test the design under various voltage and current conditions to ensure protection mechanisms work effectively.	Y
• STM32F103 should run at 72MHZ ± 5MHZ.	• Use a frequency meter to confirm the operating frequency of the STM32F103 during operation.	Y
• PCB should consume $\leq$ 1W to ensure efficient power consumption.	• Using the adjustable DC power supply, which can show the input voltage, input current, and input power, to measure the power consumption of the design to ensure it does not exceed 1W.	Y
• Capable of processing and interpreting signals from the IMU accurately.	• Put the PCB board on a horizontal table. Then, power the PCB board and run the script to check whether the IMU returns the zero point to the microcontroller. Then, rotate the PCB board 45 degrees counterclockwise 8 times and check whether the IMU returns the correct data for each rotation.	Y

Table 9: Central Control–Requirements & Verification Pt.1

Requirement	Verification	Result
• Maintaining Can bus load under 80% between the IMU and the motors to ensure stable communication.	• Run a script and Use a CAN bus analyzer to monitor the CAN bus load, ensuring it stays below 80% for stable communication.	Y
• Minimize the time used in signal processing for real-time applications to 5ms$\pm$ 5%.	• Using the inner clock to record signal processing time during the operation.	Y
• Must provide precise control signals to the motors via the CAN bus.	• Connect the PCB board and the motors and use the scripts to send the rotation signal to the motors. Check whether the motors rotate to the desired position.	Y

Table 10: Central Control–Requirements & Verification Pt.2

E.5 Payload Compartment Subsystem (3D-printed)

Requirement	Verification	Result
• The compartment should be a **10cm x 15cm x 10cm** box with covering.	• Using rules to measure the length, width, and height of the compartment to ensure the error is no more than 5%	Y
• The compartment should withstand **1-2.5lb** loads.	• Using an electronic weighing machine to measure 2.75lb objects and put them inside the compartment. Then, lift the compartment by hand and move it for 10 min. Check whether the compartment has any damage like fissures.	Y

Table 11: Payload Compartment - Requirements & Verification

E.6 Remote Controller Subsystem

Requirement	Verification	Result
• The user can use this remote control to send commands to the robot successfully.	• Using the script on the microcontroller to print the commands sent by the remote controller so that we can visualize the commands on the computer screen.	Y
• The remote control must maintain reliable communication with delay $\leq$ 10ms.	• When the microcontroller receives the data from the remote controller, we can add the time stamp at that moment. Then we can let the remote controller send the same signal continuously. The time interval between the time stamps is the communication delay. Check whether the delay $\leq$ 10ms.	Y
• The remote control must maintain an emergency kill switch to stop the robot in 2s.	• Let the robot run at a low speed and then toggle the emergency kill switch. Use a timer to record the stop time and check whether the robot is killed $\leq$ 2s.	Y

Table 12: Remote Controller - Requirements & Verification

E.7 Power System

Requirement	Verification	Result
DC-DC Converter 1: • Input: 20-27V Output: 5V ± 0.5V, $\geq$ 1A.	• Use a multimeter to measure input and output voltage and current to ensure they are within specified limits.	Y
DC-DC Converter 2: • Input from Converter 1 Output: 3.3V ± 0.3V, 500mA ± 5%.	• Use a multimeter to measure input and output voltage and current to ensure they are within specified limits.	Y
Battery: • Supply: 20-27V, 20A peak, 5A continuous Runtime: $\geq$ 30 minutes at 5A.	• Use a multimeter to measure the output voltage and current from the battery under load. Use a timer to ensure it lasts for at least 30 minutes at a 5A load.	Y
Safety and Flexibility: • Quick power cut-off: $\leq$ 200ms Switch between battery and wired power.	• Test the cut-off response time with a stopwatch or electronic timer. Check the seamless switching between power sources under operation.	Y

Table 13: Power System - Requirements & Verification

Appendix F Simulink Simulation Block Diagram

F.1 Wheel Motion Simulation Block Diagram

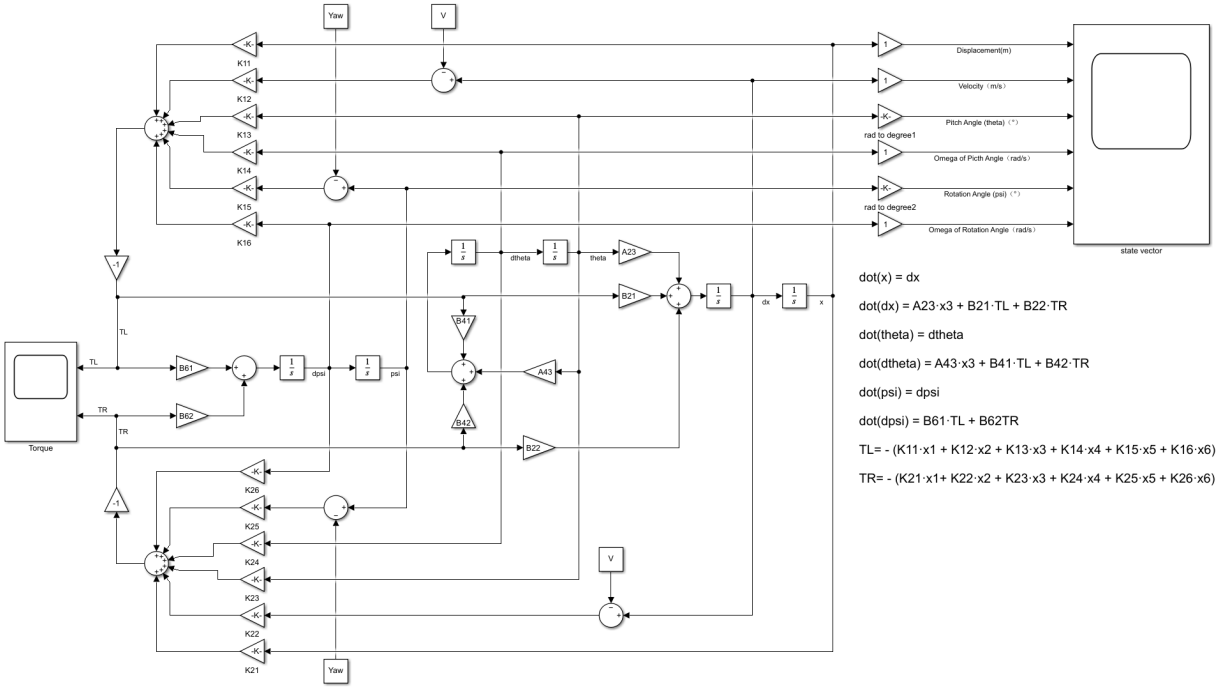

Figure 31: Simulink Wheel Model

F.2 Leg Motion Simulation Block Diagram

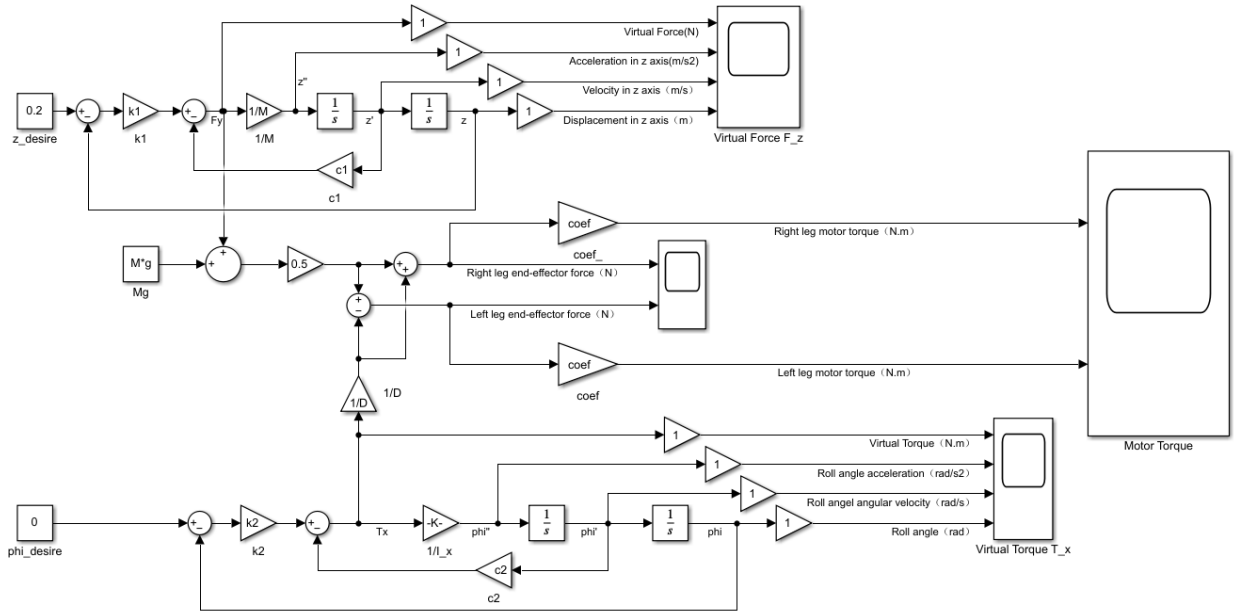

Figure 32: Simulink Leg Model

Appendix G Simulation Test Result

G.1 Wheel Motion Simulation Test

G.1.1 Velocity Simulation Test

We set the target speed to 0.5 m/s, and use the Simulink to simulate the robot behavior. The state vector simulation result is shown in **Figure 33** and the input vector simulation result is shown in **Figure 34**.

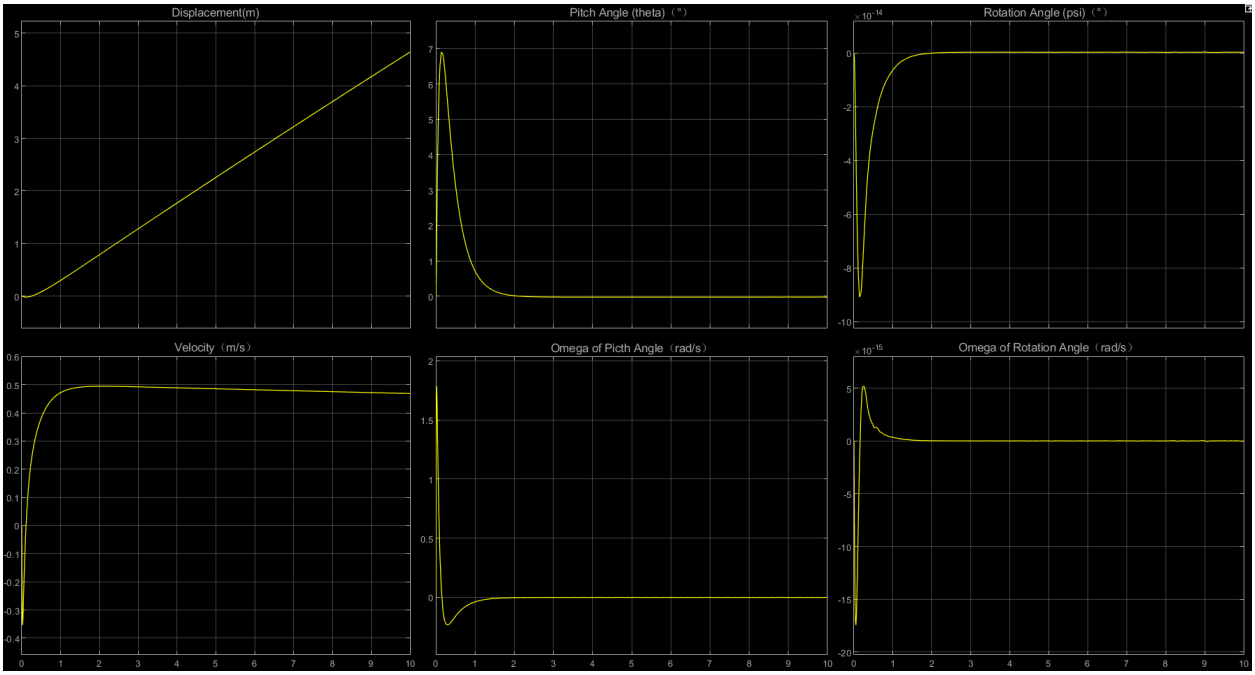

Figure 33: State Vector Simulation Result

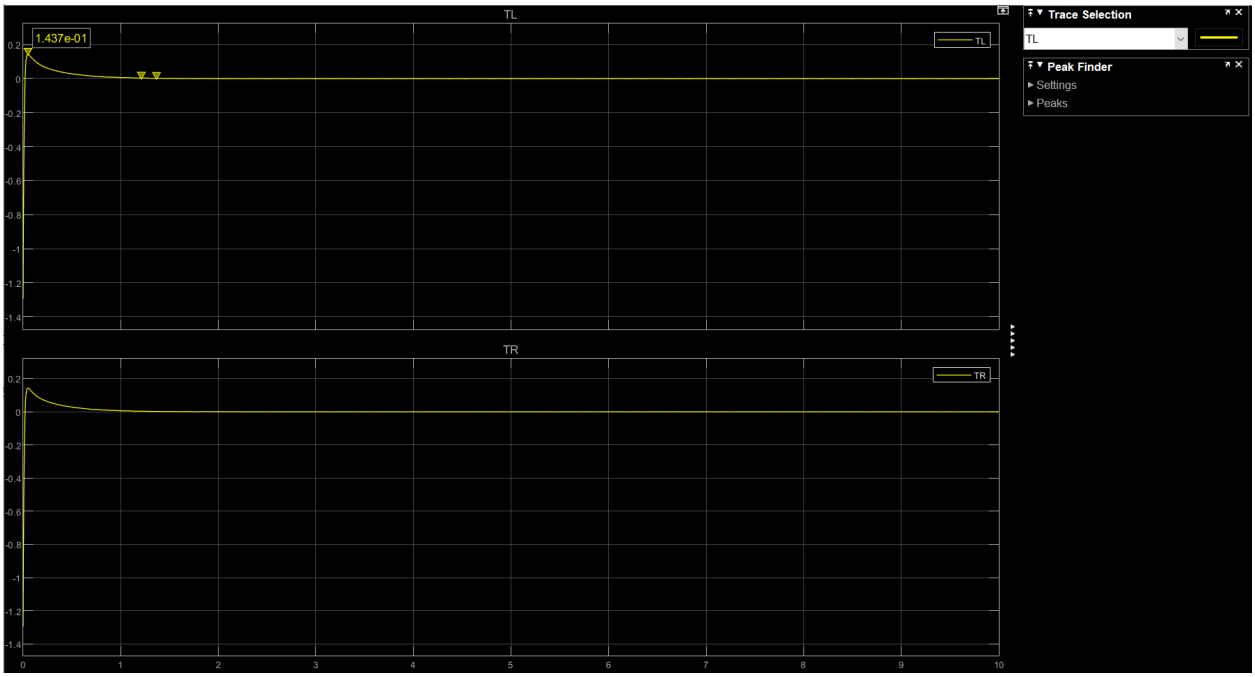

Figure 34: Input Vector Simulation Result

The robot uses about 1 s to reach the target speed of 0.5 m/s. During this process, the maximum pitch angle is about 7 degrees and then converges to 0. Meanwhile, the rotation angle is pretty small, and the peak torque of each wheel motor is 0.1437 Nm within the peak torque range of the selected motor, which matches our requirements.

G.1.2 Rotation Simulation Test

We set the target rotation angle to 30 degrees, and use the Simulink to simulate the robot behavior. The state vector simulation result is shown in **Figure 35** and the input vector simulation result is shown in **Figure 36**.

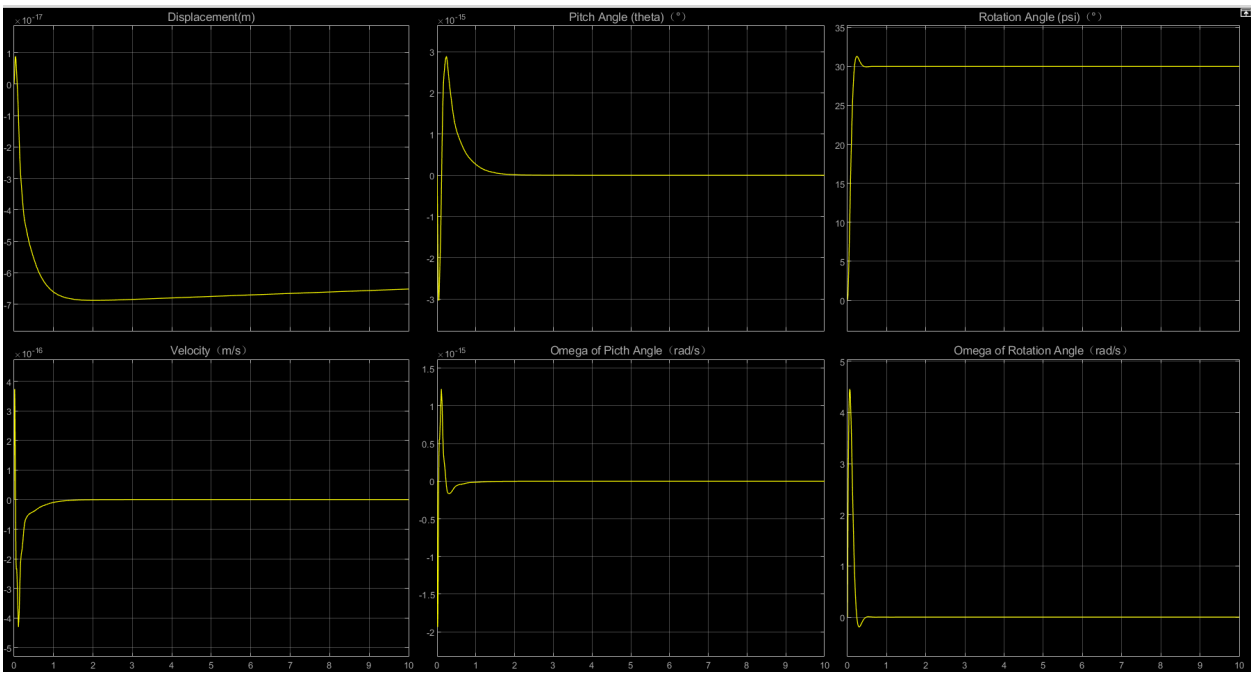

Figure 35: State Vector Simulation Result

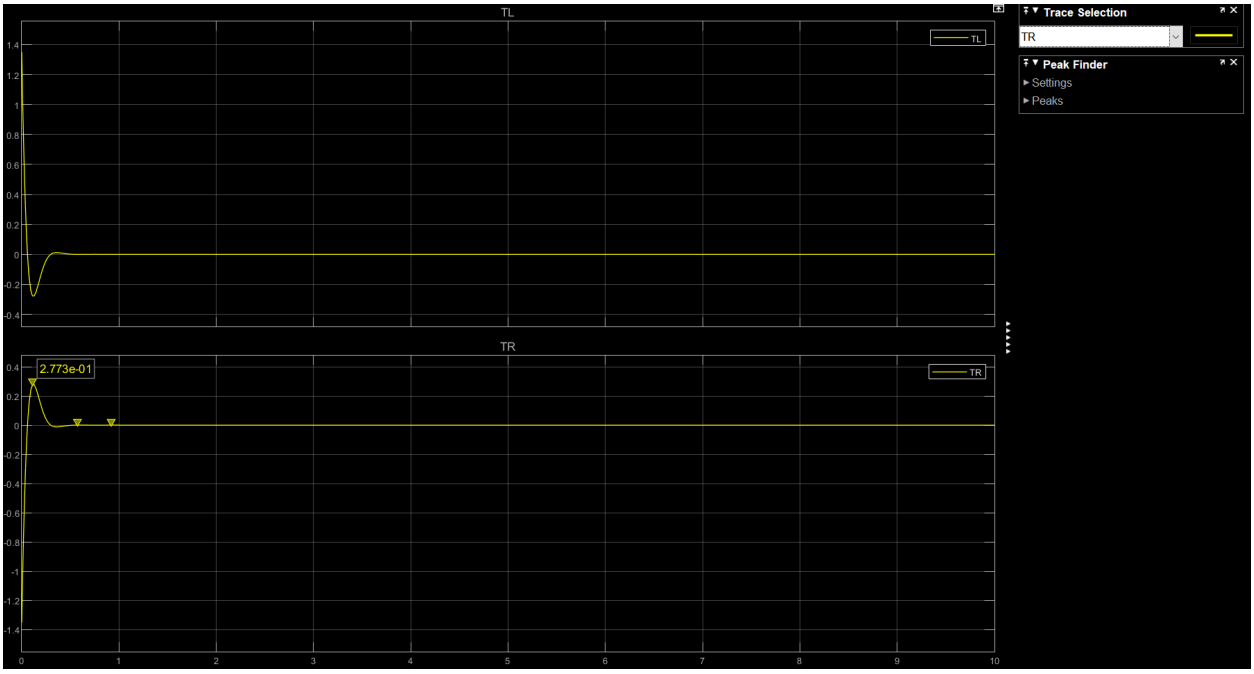

Figure 36: Input Vector Simulation Result

The robot uses less than 0.5 s to reach the target rotation angle of 30 degrees. During this process, the maximum pitch angle omega, velocity, and displacement are almost 0.

Meanwhile, the peak torque of each wheel motor is 0.2773 Nm within the peak torque range of the selected motor, which matches our requirements.

G.2 Leg Motion Simulation Test

G.2.1 Elevation Simulation Test

We set the target elevation height to 0.2 m, and use the Simulink to simulate the robot behavior. The variable simulation result is shown in **Figure 37** and the motor torque simulation result is shown in **Figure 38**.

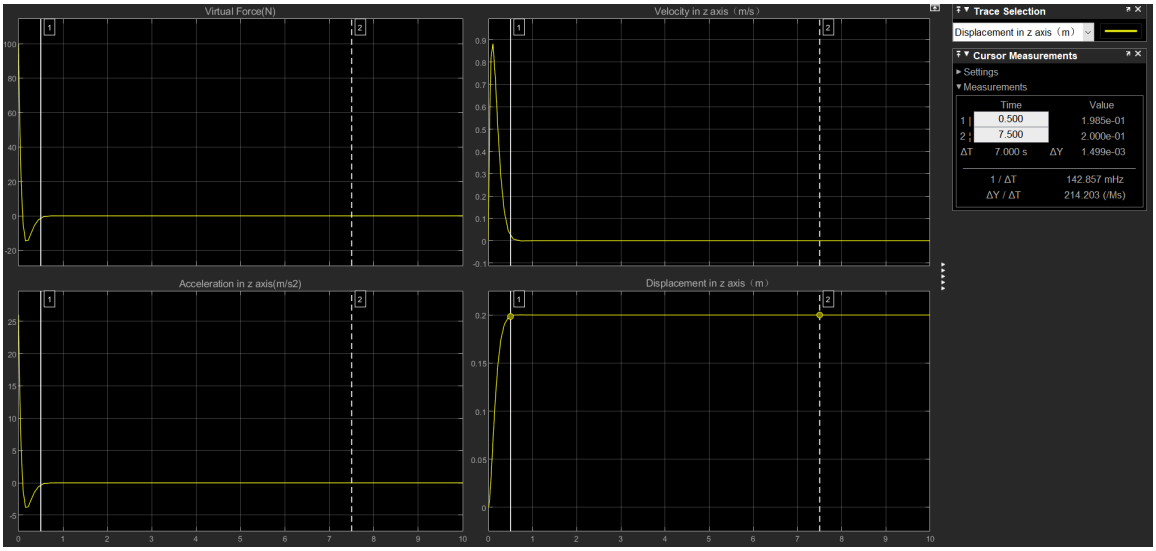

Figure 37: Variable Simulation Result

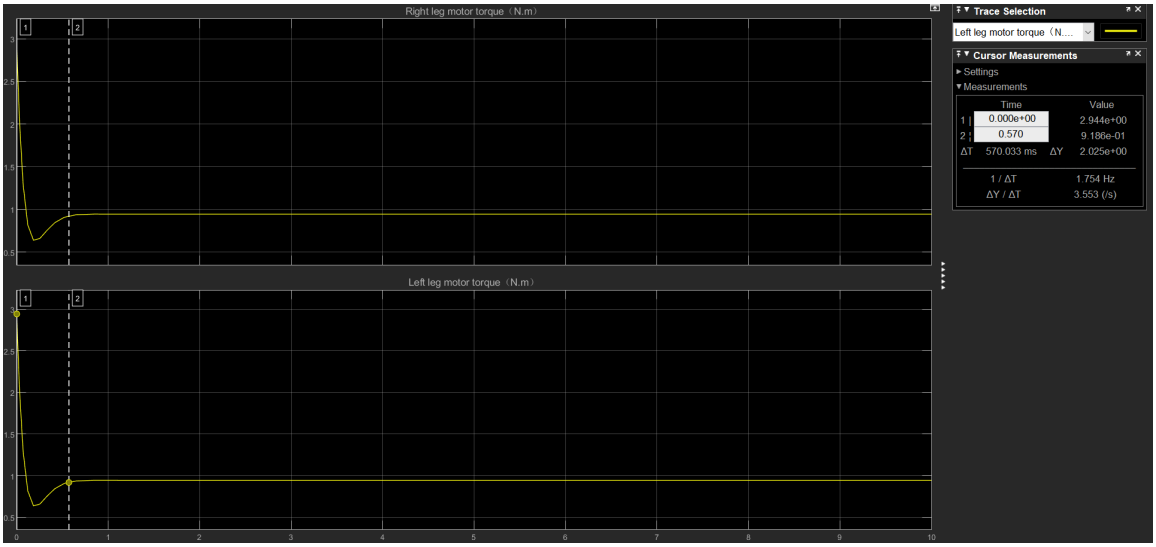

Figure 38: Motor Torque Simulation

The robot uses about 0.5 s to reach the desired height. During this process, each motor's maximum torque is about 3 Nm within the peak torque range of the selected motor, which matches our requirements.

G.2.2 Adaptive Suspension Simulation Test

We set the initial roll angle be 0.2 rad and use the Simulink to simulate the robot behavior. The variable simulation result is shown in **Figure 39** and the motor torque simulation result is shown in **Figure 40**.

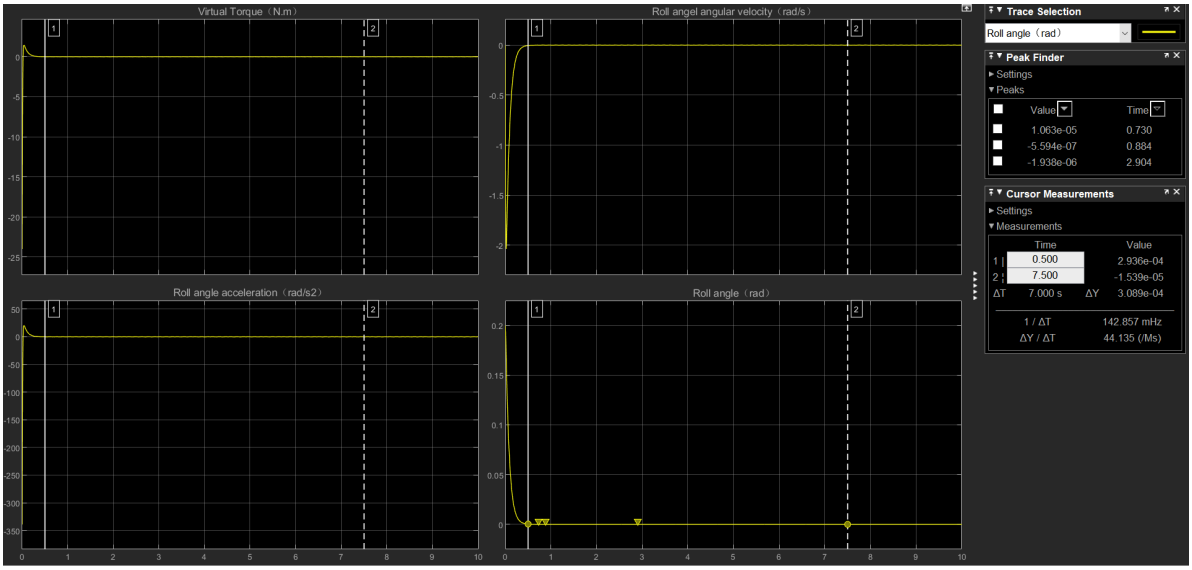

Figure 39: Variable Simulation Result

Figure 40: Motor Torque Simulation

The robot uses about 0.5 s to come back to the balance state. During this process, the left motor's maximum torque is about 4.4 Nm, and the right motor's maximum torque is about 2.5 Nm within the peak torque range of the selected motor, which matches our requirements.

Appendix H Cost Table and Schedule

H.1 Cost Table

Comment	Designator	Manufacturer	Quantity	Price	
100nF Unpolarized capacitor	C1,C2,C3,C4	Walsin Tech Corp	4	0.571	2.284
10nF Unpolarized capacitor	C5, C6	Walsin Tech Corp	2	0.0098	0.0196
1uF Unpolarized capacitor	C7,C8,C9	Walsin Tech Corp	3	0.0267	0.0801
10pF Unpolarized capacitor	C10,C11	Samsung Electro-M	2	0.0214	0.0428
22uF Unpolarized capacitor	C12,C13	Murata Electronics	2	0.2103	0.4206
100nF Unpolarized capacitor	C14,C15	Knowles	2	0.3441	0.6882
RED Light emitting diode	D1	Hubei KENTO Elec	1	0.0051	0.0051
120R Ferrite bead	FB1	Sunltech Tech	1	0.0104	0.0104
Conn_01x04_Pin Generic connector	J1,J2,J4		3	0.1077	0.3231
USB_C_Plug_USB2.0	P1	DEALON	1	0.0516	0.0516
PMBT3904YS 200mA IC	Q1	Nexperia	1	0.0534	0.0534
10k Resistor	R1,R6	Viking Tech	2	0.0569	0.1138
1k5 Resistor	R2,R3,R4,R5	Viking Tech	4	0.0012	0.0048
4k7 Resistor	R9	Viking Tech	1	0.0025	0.0025
SW_SPDT Switch	SW1	DongGuan KINGTEK	1	0.3409	0.3409
SW_Push Push button switch	SW2	C&K	1	0.4437	0.4437
STM32F103C8T6	U1	STMicroelectronics	1	1.418	1.418
AMS1117-3.3	U2	Advanced Monolithi	1	0.1399	0.1399
MAX3051	U3	Analog Devices Inc./	1	0.91	0.91
BMI088	U4	Bosch Sensortec	1	2.9489	2.9489
16MHz Four pin crystal	Y1	TXC Corp	1	0.1764	0.1764
DM-J4310-2EC		DAMIAO Tech	2	164.37	328.74
M3508		DJI	1	68.49	68.49
C620 ESC		DJI	1	54.65	54.65
100C 1000mAh 22.2V Lipo Battery		OVONIC	1	23.99	23.99
XT60 Plug		AMASS	1	0.3032	0.3032
HATCHBOX PLA PRO+ 3D Printer Filament		Hatchbox	1	28	28
I2C 0.96-inch OLED Display?	AZDelivery		4	1	4
GH1.25 Connector		YouXin Electronics	20	0.2	4
GH1.25 Cable 2P/15CM		YouXin Electronics	5	0.5	2.5
NCE60P25K		YouXin Electronics	5	0.3	1.5
NCE6020AK		YouXin Electronics	5	0.2	1
XT60 PW-M		YouXin Electronics	5	0.5	2.5
XT30UPB-F		AMASS	20	0.25	5
Emergency STOP Button 16mm		BaoLing Electronics	1	5.99	5.99
Aluminum Stand M3*12mm		HKM shop	30	0.5	15
AXK0821 bearing		yxvsy	20	1.5	30
F688ZZ		yxvsy	20	0.3	6
Carbon Fiber Plate		Hcw Fiber	1	209	209
		Total:			801.141

Figure 41: Bill of materials for the entire robot

Note: Battery, Motors, and remote controller are borrowed from RSO: Illini Robomaster. The prices for these components are not listed in the table.

H.2 Schedule

Week	Task	Person
2023/9/25-2023/10/01	Modify Project Proposal and Prepare for Design Document	Everyone
	Do wheeled motion(forward and backward) analysis	Gabriel
	Finish up PCB design	Jerry
	Complete mechanical structure modeling	Zehao
2023/10/02-2023/10/08	Prepare for Design Review	Everyone
	Do wheeled motion(rotation) analysis	Gabriel
	Pin Configurations in STM32CubeIDE	Jerry
	Testbench design and build, finish power-pcb design	Zehao
2023/10/09-2023/10/15	First Round of PCB Order	Everyone
	Do Leg motion(support) analysis	Gabriel
	Construct drivers for each component (Motors, IMU ...)	Jerry
	Assemble the mechanical parts, design and print out payload parts	Zehao
2023/10/16-2023/10/22	Second Round of PCB Order	Everyone
	Do Leg motion(jump) analysis	Gabriel
	Test functionality of individual motors with an existing development board & review PCB design	Jerry
	Connect the structure to the test bench and revise design	Zehao
2023/10/23-2023/10/29	Individual Process Report & Third Round of PCB Order	Everyone
	LQR regulator implementation	Gabriel
	Test PCB and make changes if necessary	Jerry
	Modeling a complete wheeled legged robot	Zehao
2023/10/30-2023/11/05	Check progress of everyone and help each other if needed	Everyone
	Implement VMC Control Algorithm	Gabriel
	Pin configuration and code deployment on PCB	Jerry
	Making parts and sending CAD files to factory	Zehao
2023/11/06-2023/11/12	Check progress of everyone and help each other if needed	Everyone
	Remote controller communication	Gabriel
	Integrate PCB into the robot	Jerry
	Assembling whole robot	Zehao
2023/11/13-2023/11/19	Mock Demo and Team Contract Fulfillment	Everyone
	Moter driver implementation	Gabriel
	Test and revise the PCB board	Jerry & Zehao
2023/11/20-2023/11/26	Integrating all parts and test the code functionality	Everyone
2023/11/27-2023/12/03	Final Demo	Everyone
2023/12/04-2023/12/07	Final Presentation and Final Paper	Everyone

Figure 42: Detailed Schedule