DEVELOPMENT AND CONTROL OF A STAIR-CLIMBING VACUUM CLEANER ROBOT

by

Kanyasorn Phongphaitoonsin

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Engineering in Mechatronics and Machine Intelligence

Examination Committee: Prof. Manukid Parnichkun (Chairperson)
Prof. Huynh Trung Luong
Dr. Mongkol Ekpanyapong

Nationality: Thai
Previous Degree: Bachelor of Engineering in Mechatronics
Engineering Assumption University
Bangkok, Thailand
Scholarship Donor: His Majesty the King's Scholarships
(Thailand)

Asian Institute of Technology
School of Engineering and Technology
Thailand
May 2024

AUTHOR'S DECLARATION

I, Kanyasorn Phongphaitoonsin, declare that the research work carried out for this thesis was in accordance with the regulations of the Asian Institute of Technology. The work presented in it are my own and has been generated by me as the result of my own original research, and if external sources were used, such sources have been cited. It is original and has not been submitted to any other institution to obtain another degree or qualification. This is a true copy of the thesis, including final revisions.

Date: 7th May 2024

Name (in printed letters): KANYASORN PHOGNPHAITOONSIN

Signature: Kanyasorn Phongphaitoonsin

ACKNOWLEDGMENTS

First and foremost, I extend my utmost respect and gratitude to my advisor, Prof. Manukid Parnichkun, for his invaluable guidance, suggestions, and unwavering encouragement throughout the entirety of this research. Without his expertise and advice, completing this thesis would have been significantly more challenging.

I wish to express my sincere appreciation to Prof. Huynh Trung Luong and Dr. Mongkol Ekpanyapong, the esteemed members of the examination committee, for their insightful remarks and generous sharing of knowledge, which greatly enriched the quality of this work.

Special thanks are due to Mr. Hoang Hung Manh and Mr. Thanit Pattana, the technical staff of the Mechatronics Department laboratory, whose dedicated assistance played a crucial role in the successful completion of the robot prototype.

I am deeply grateful to the His Majesty the King's Scholarships of Thailand for generously funding my entire master's degree program at AIT, enabling me to pursue my academic aspirations.

Lastly, I extend my heartfelt thanks to my family for their unwavering support and encouragement throughout my educational journey. Their love and assistance have been indispensable in my pursuit of academic excellence.

ABSTRACT

This thesis addresses the limitations of existing stair-climbing mechanisms, which are often characterized by large size, complexity, or high cost. Introducing a novel approach, a vacuum cleaner robot tailored for stair climbing is proposed and developed herein. Utilizing a scissor lift design, the robot incorporates three legs strategically weighted for enhanced stability through center of gravity analysis. Each leg is equipped with a stepper motor for precise control, while mecanum wheels, actuated by N20 gear motors, facilitate omnidirectional movement. The sensing and control system employ low-cost sensors, specifically ultrasonic and microswitch, to detect stair edges. Experimental evaluation encompasses movement, turning, and ascent/descent timing. While movement experiments reveal errors attributed to the six-wheel design, turning angles remain acceptable despite positional shifts. Optimal stepper motor control is determined at 500 steps per second, synchronized with the speed of N20 gear motors across all experiments. Notably, the implemented sensing and control system demonstrate successful stair climbing and fall prevention during descent.

Keywords: vacuum cleaner robot, scissor lift design, lead screw, ascending, descending stair

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Problem Statement

I opt to delegate today's arduous and monotonous cleaning duty to the robot, allowing it to autonomously tackle the task. Notably, the global cleaning robot market exhibited remarkable growth, with a compound annual growth rate (CAGR) of 24.3%, escalating from $9.24 billion in 2022 to $11.49 billion in 2023. Projections from (ReportLinker, 2023) suggest that the cleaning robot market is poised for further expansion, with an anticipated value of $27.06 billion by 2027, driven by a CAGR of 23.9%.

Numerous sought-after products currently dominate the market, particularly robots characterized by their nonmodular or fixed circular design. This design enables them to efficiently navigate predefined floor areas and autonomously map their surroundings using onboard sensors, effectively cleaning smooth surfaces. However, a limitation of these conventional robots is their inability to clean staircases, a crucial feature in many residential and commercial settings, as they are optimized solely for smooth, floor-like surfaces.

Figure 1.1

Example of Marketable Robot Vacuum Cleaner

Numerous studies have been conducted on robot vacuums from various angles. The mechanism design, which includes the robot's shape and intended task, comes first (Iwan R. Ulrich, 1997). To pinpoint the position of the robot and other objects on the map, navigation and localization techniques are employed, as outlined by Hartwell (2022). Third, the robot's autonomy or path planning, which is used to create an efficient and effective walking path (B.K. Patle, 2019).

Despite the fact that there are many studies and robots available, as was already mentioned, the majority of them only focus on cleaning the flat and pay little attention to the staircase, which is an important part of the building.

This is an illustration of earlier research in this field. The mechanism was developed by sTetro (Ramalingam, et al., 2021) and uses a vertical conveyor belt to lift the robot up and across stairs in addition to allowing it to move on flat surfaces. A brand-new, small stair-cleaning robot scales stairs by rotating and moving its leg in different directions (Zhang, 2016). the robot's L-shaped legs, which it uses to move and ascend (T. Kakudou, 2011).

This study aims to devise a sophisticated mechanism for the robot, leveraging low-cost sensors like ultrasonic and switch technology to adeptly detect stairs, thereby enhancing its navigational capabilities with ingenuity and cost-effectiveness.

1.2 Objectives of The Research

The primary goal of this thesis revolves around crafting a vacuum cleaner robot capable of ascending stairs. To attain this objective, the following sub-goals are imperative:

- Crafting a novel mechanism tailored specifically for a vacuum cleaner robot.

1.3 Scope

The scope of this thesis is as follows:

1. The robot is capable of climbing the stairs 10 cm of height.
2. The robot is capable of climbing the stairs 38 cm of width.
3. The robot detects the obstacle using ultrasonics and microswitch.
4. The robot is planned to have a height of less than 20 cm.
5. The robot will move at a 10 cm/s speed.

1.4 Contribution

The conveyor belt, L- or Z-shaped robot legs are the only parts of prior research that have been examined thus far. Therefore, the goal of this research is to enhance the robot's mechanism so that it can climb stairs and move across flat surfaces.

CHAPTER 2
LITERATURE REVIEW

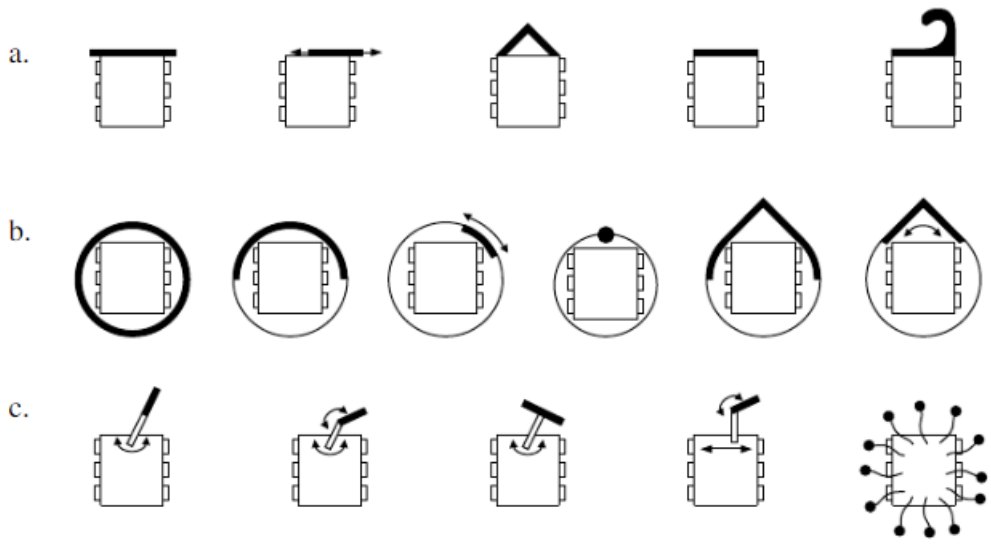
This chapter explores the research behind creating and deploying vacuum cleaner robots. It covers various techniques for designing their mechanisms and algorithms, providing valuable insights into their development and practical application.

2.1 Mechanism Design

2.1.1 Design of Vacuum Cleaner Robot

Figure 2.1

Shape of Vacuum Cleaner Robot



An autonomous vacuum cleaner 's ability to cover a surface, which is its primary duty, is significantly influenced by the design of the robot. Three categories can be used to group the potential shapes of autonomous vacuum cleaners: simple shapes, spherical robots, and robots with an arm.

All the simple forms are simple to construct. But none of them can effectively clean corners around objects like legs or require a complicated trajectory to clean along walls. The advantage of the round robots is that they make it easier to navigate around obstructions. But a key drawback for potential markets is that spherical robots cannot

clean corners. Additionally, they need a complicated trajectory to clean around little things like legs.

According to the research, a robot that can climb stairs could be built in a simple shape because stairs typically range in width from 10 to 30 cm. The round robot can be used, but it might be more challenging to get it to clean the corner of the stairs. (Iwan R. Ulrich, 1997)

2.1.2 Types of Stairs Climbing Robot

1. **Biped Robot**

Figure 2.2

Example of Biped Robot

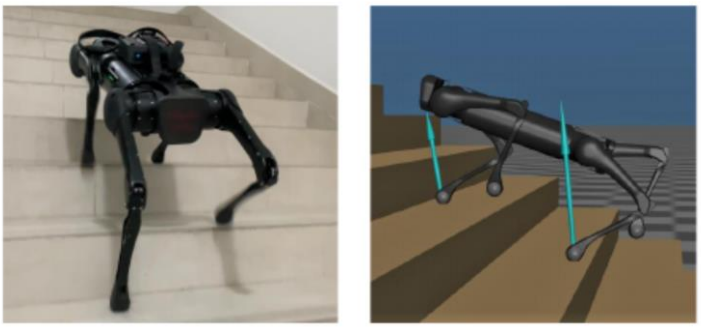
This paper delves into the exploration of motion control for a biped robot capable of navigating both ascending and descending staircases. With the specified stable margin and walking pace, the algorithm constructs walking patterns. All that is needed is the rough stair height and breadth.

This algorithm determines the ideal hip height and generates the hip and foot trajectories using a cubic polynomial. Continuous dynamic walking up and down stairs is achievable when the initial and finish speeds of a walking cycle are controlled. Simulation demonstrates that this technique can perform dynamic stair climbing or descending. A 12 DOF biped robot called RoboSapien is created to test the walking algorithm. (Vadakkepat, 2003)

2. Quadruped

Figure 2.3

Example of Quadruped

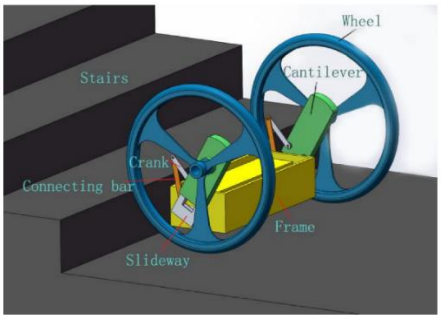


This article explains the method that will allow a robot to carry a load up stairs. Its mechanical layout makes it appropriate for stair climbing with front and back wheels powered by DC motors. Up until recently, stair climbing robots were built with a lot of hardware and used a chain roller to help them go up and down steps or across a flat surface. This robot's mechanical design uses fixed and flexible links of wheel legs rather than chain rollers that move in relation to one another to create high friction with steps. (Jindal, 2015)

3. Wheel-Leg

Figure 2.4

Example of Wheel-Leg Robot



The primary structure of the robot comprises the frame, along with symmetrical telescoping cantilevers on the left and right sides, in addition to the wheel assembly. A motor, situated at the lower end of the telescopic cantilever, facilitates adjustment of

the frame's orientation to ensure horizontal stability during movement. At the top of the telescoping cantilever resides the main drive motor responsible for propelling the wheels. Within the telescopic cantilever, a telescopic mechanism creates a translation joint. The robot's configuration follows a "4R+2P" design, consisting of four rotational joints and two translation joints, which govern the movement of the frame, left and right telescoping cantilevers, and the left and right wheels, enabling its functionality as a two-wheeled ascending stair robot. (Wu, 2022)

4. **ASGUARD**

Figure 2.5

Example of ASGUARD Robot

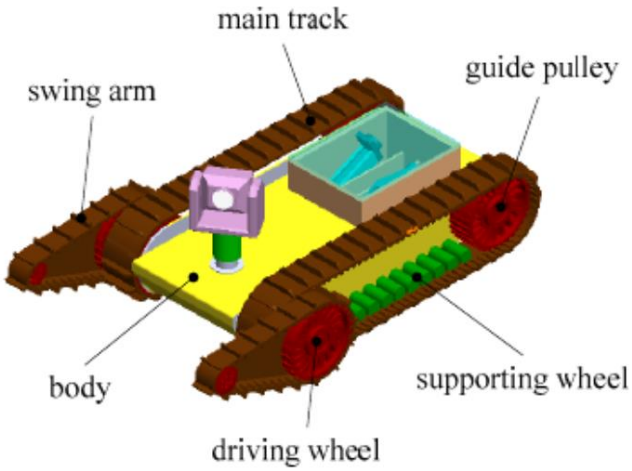
Eich et al. (2008) conducted an evaluation of the ASGUARD robot, a unique hybrid crawler-type equipped with both legs and wheels, tailored for operations in rugged outdoor environments such as security, outdoor surveillance, and disaster response missions. These missions, often referred to as "Three D" missions—short for dull, dangerous, and dirty—require robots to navigate challenging terrains while carrying an array of mission-specific sensors.

In scenarios where the terrain is relatively flat and straightforward to traverse, the robot demonstrates swift mobility. The robot's locomotion is facilitated by multiple rotating legs that move along a single hip shaft. Both (Eich et al., September 8–10, 2008) and (Eich et al., January 7-8, 2008) delve into the conceptual framework and design aspects of the ASGUARD robot. (Eich, 2009)

5. Track Robot

Figure 2.6

Example of Track Robot



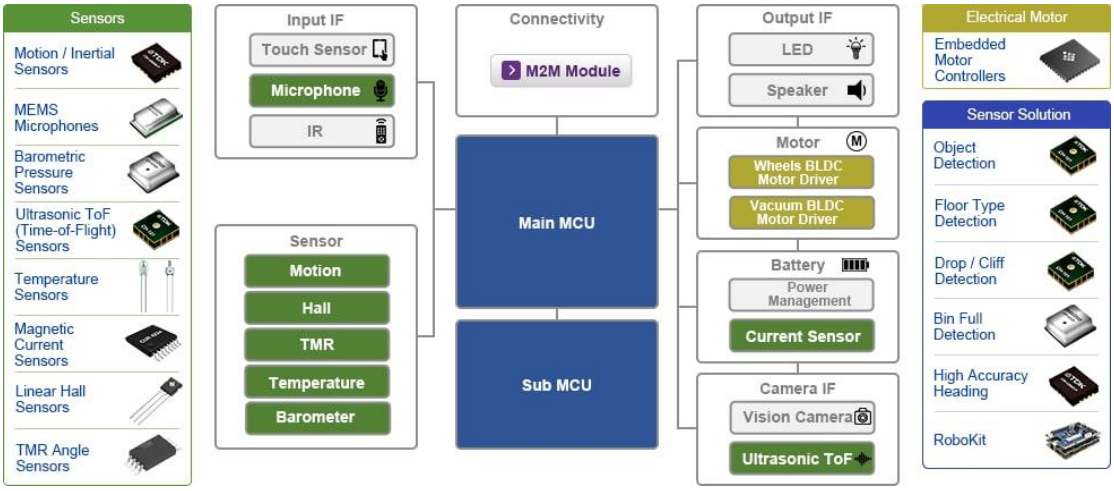
This study focuses on the development of a tracked mobile robot specifically designed for search and rescue operations within nuclear plants. Analysis is done on the robot's stair climbing abilities. The control system's hardware and software are constructed, and an interface for human-computer interaction is created. The robot's dynamic model is created, and its trajectory tracking control algorithm is described. The simulation tests using the Octave program are now complete. Simulation results validate the efficacy of the proposed method, indicating that the suggested control algorithm enables the tracked robot to achieve exceptional trajectory tracking performance on both linear and cambered paths. (Dong, 2016)

2.2 Navigation and Localization

2.2.1 Sensor in Vacuum Robot

Figure 2.7

Sensor in Vacuum Robot (Hartwell, 2022)



1. **Ultrasonic Time-of-Flight Sensors**

Ultrasonic time-of-flight (ToF) sensors provide precise range measurements in any lighting condition, irrespective of the target's color or optical transparency. With a broad field of view (FoV), these sensors can simultaneously measure the ranges of multiple objects. In robot vacuums, they play a crucial role in detecting obstacles such as pets or toys, allowing the robot to adjust its path accordingly to avoid collisions.

2. **Short-Range Ultrasonic Tof Sensors**

Short-range ultrasonic time-of-flight (ToF) sensors can distinguish between various types of flooring surfaces. By analyzing the average amplitude of reflected ultrasonic signals, the application can determine whether the target surface is soft (like carpet) or hard (like hardwood). In the context of a robot vacuum, this capability allows for adaptive motor control: the vacuum can adjust its motor speed accordingly, slowing down when transitioning from carpet to hardwood floors, as less effort is required for cleaning on the latter surface.

The cliff detection feature in a robot vacuum serves to identify when it reaches the edge of a flight of stairs, thus preventing potential falls.

3. VSLAM and Lidar

To generate a virtual map of a room, many companies manufacturing high-end robot vacuums rely on either visual simultaneous localization and mapping (VSLAM) or LiDAR technology. These advanced technologies enable the robot vacuum to navigate more efficiently and clean entire levels of a house with multiple rooms. However, if the robot vacuum is lifted and placed in a new location, it may not recognize its surroundings immediately. In such cases, the robot must explore its environment in various directions to determine its location. Once it detects objects and starts mapping the walls, it can deduce its position relative to the existing map.If the robot vacuum moves behind a table or couch, for example, where there is less light, VSLAM or LiDAR technology may not be suitable.

4. Inertial Measurement Units (IMU)

Inertial Measurement Units (IMUs) capture the roll, pitch, and yaw of the robot vacuum's movements in both linear and rotational dimensions within the physical world. This data aids the robotic vacuum in maintaining its intended trajectory, whether it's moving in a straight line or following a circular path. Even in cases where there's a minor discrepancy between the intended and actual positions, the IMU ensures accurate tracking of the robot vacuum's location.

The robot vacuum leverages both rotational and linear movement data, along with room mapping, to discern whether it's revisiting areas it's already cleaned. Furthermore, it possesses the capability to resume its cleaning progress seamlessly in the event of a battery depletion. Even if manually relocated or reoriented, the robot vacuum can discern its new position and orientation within the physical space. The IMU plays a pivotal role in ensuring the efficiency and effectiveness of robot vacuums in navigating and cleaning environments.

Dead reckoning, a technique that integrates data from wheel revolutions, inertial measurements from the IMU, and object recognition from Time-of-Flight (ToF) sensors, can be employed to estimate the position and navigate robot vacuums that don't rely on Visual Simultaneous Localization and Mapping (VSLAM) or Light Detection and Ranging (LiDAR) mapping technologies.

5. Smart Speaker Microphones

Microphones are increasingly integral in the evolution of robot vacuums, especially as manufacturers integrate artificial intelligence (AI) and voice assistant capabilities. Beamforming, a radio frequency (RF) management technique, is particularly noteworthy. With AI-driven refinement, beamforming directs the microphone towards the desired signal while minimizing noise interference. Although current robot vacuums emit some noise from their motors and rotating brushes, advancements in beamforming and microphone technologies promise quieter operations in the future. Eventually, microphones may even discern and respond to user commands amidst background noise.

Algorithms can be trained to filter out extraneous noises and focus solely on detecting the user's speech. In practice, users may desire direct control over the vacuum cleaner without relying on an app or voice assistant. This functionality could be executed within the robot vacuum's onboard processor in real-time. Alternatively, upon detecting speech through the microphone, the robot vacuum could pause all motor functions and attend to the received command.

6. Embedded Motor Controllers

To ensure precise movement and accurate directionality, the embedded motor controllers in the robot vacuum utilize gears to monitor and control the rotation of the wheels. These motor controllers are equipped to detect even minor deviations, distinguishing between subtle changes such as a 90-degree turn versus an 88-degree one. Without this great degree of accuracy, the robot vacuum will eventually veer off course. The robot vacuum can be scaled up or down depending on whether sensors are used because the embedded motor controller is adaptable.

7. Pressure Sensors

To gauge the quantity of dust accumulated within the dustbin, a pressure sensor measures the airflow passing through it. As dust accumulation or filter blockage increases suction, airflow becomes stagnant, causing a decline in air pressure inside the dustbin compared to its initial state when empty. However, for more accurate detection, it's recommended to measure this as a differential pressure. This involves employing a

pressure sensor to assess both the air pressure inside the dustbin and the ambient air pressure outside, enabling more precise detection of changes.

Many high-end bases come equipped with the capability to automatically suction out the contents of the dust box, providing a convenient and hands-free method for maintaining the cleanliness of the robot vacuum. After returning to its starting point and emptying its contents, the robotic vacuum can resume cleaning.

8. **Auto-Recharging**

You require precise current and voltage measurements to calculate the battery's state of charge (SoC). This data is provided by the NTC thermistors and coulomb counters in the battery pack.

When the battery of the robot vacuum reaches a predetermined state of charge (SoC), it triggers a command to cease cleaning and return to the charging base for a recharge. The robot vacuum then navigates back to its last known location and resumes cleaning once its battery is fully charged, ensuring uninterrupted cleaning sessions. No of the size of the room, the robot vacuum should be able to clean the entire floor area with different charging and garbage emptying options.

9. **Thermistors**

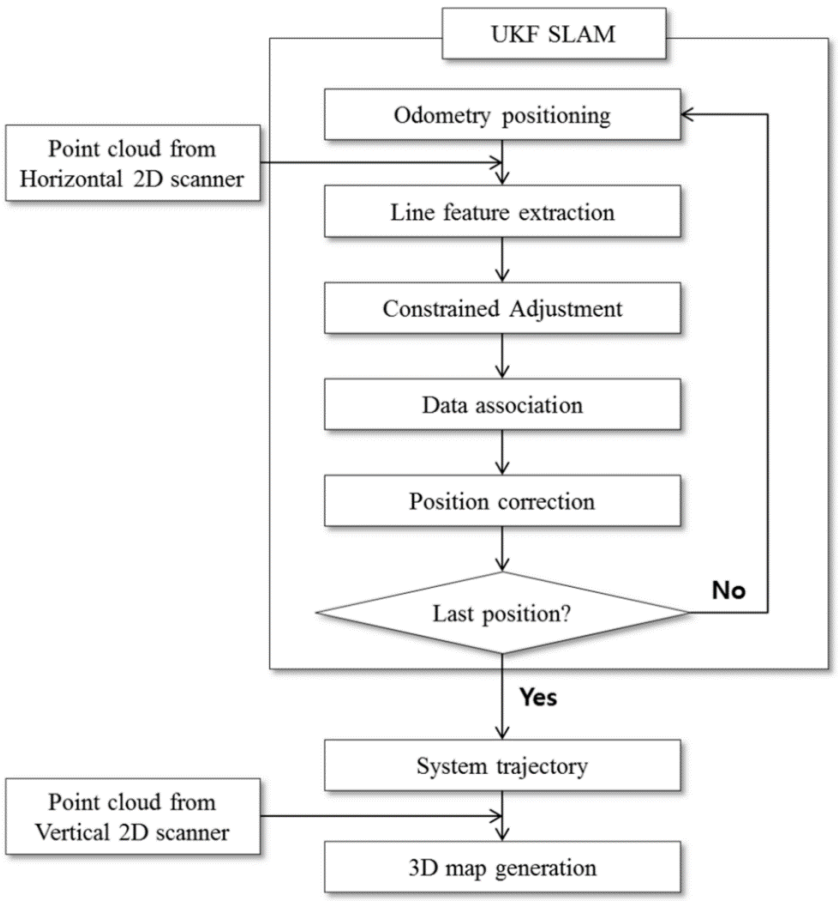
Temperature sensors, such as thermometers, are utilized to monitor the operating temperature of the Microcontroller Unit (MCU) or Microprocessor Unit (MPU) in the robot vacuum. Additionally, these sensors can assess the temperatures of the motors and brush gears. If these components operate at excessively high temperatures, the robot vacuum is prompted to pause and potentially conduct system diagnostics. This precautionary measure ensures that the robot vacuum operates within safe temperature limits, preventing potential damage or malfunction.Additionally, objects like an elastic band or extra hair that get caught in the brushes might cause the motors to overcompensate and overheat.

2.2.2 Simultaneous Localization and Mapping (SLAM)

Due to the presence of occlusions, the traditional static laser scanning approach has some limitations regarding its operability in complicated indoor environments. Surveyors often need to reposition the scanner multiple times to capture complete scans of interior spaces, resulting in additional effort to register each scanned point cloud. As an alternative approach, a kinematic 3D laser scanning system is proposed here, which continuously maps using the Simultaneous Localization and Mapping (SLAM) technique based on line features. Moreover, to reduce the uncertainty of line-feature extraction, a limited correction method is introduced. This correction is based on the assumption that primary structures in typical interior environments exhibit parallel or orthogonal line features. (Jung J, 2015)

Figure 2.8

Flowchart of 3D SLAM



2.3 Path Planning for Vacuum Cleaner Robot

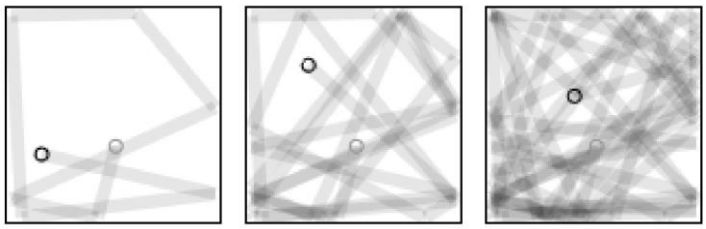
2.3.1 Basic Algorithms (Sörme, 2018)

1. Algorithm Based on Random Walk

Random algorithms primarily generate random numbers, requiring minimal hardware or processing power from the robot. Random-based techniques, however, have no guarantee of cleaning performance within a specific amount of time because modern robots cannot run indefinitely (they must run out of power eventually). In figure 2.9, the robot is denoted by a black circle, while the initial position is represented by a gray circle. Direction changes persist throughout the walk, with intensity indicating frequency of robot traversal.

Figure 2.9

Random Walk Algorithm Simulation

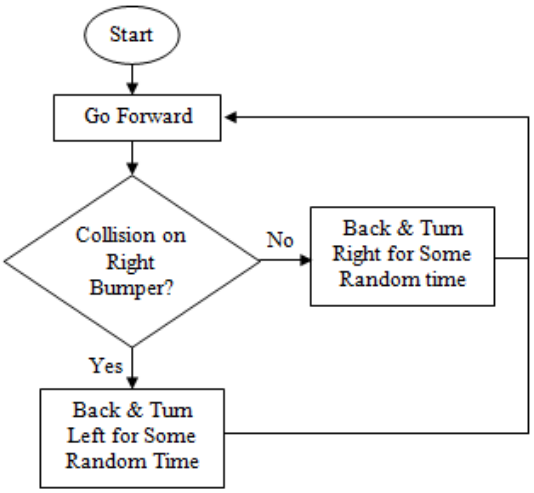


Robots are thought to use randomized path planning algorithms when they randomly choose their course. These algorithms can be completely arbitrary. For instance, in Figure 2.9, the robot randomly chooses a new heading after each collision with an object, paying no attention to its previous or current location. Developers can then add special case handling to this crude strategy, such as handling situations when users are stuck in tight spaces or corners.

In a random walk-based algorithm, sensors can detect obstacles and determine whether the robot should turn left or right when encountering them (for example, if an obstacle is detected on the left, the robot will turn right). By continuously turning away from obstacles in this manner, the robot can navigate around them. The actual turning direction is determined by generating a final random number. With this approach, the robot moves in a way that is very dependent on its surroundings.

Figure 2.10

Flowchart of Random Walk Based Algorithm



2. Spiral Path Algorithm

The Spiral Path Algorithm combines spiral patterns with random turns. Similar to the previously mentioned random walk-based algorithm, this approach involves the robot creating a spiral path when it determines sufficient space. Onboard sensors are used to make this determination. If there is ample space, the robot initiates continuous turns with increasing radius from its current position, forming a spiral trajectory.

This might be accomplished on a robot with two main wheels by progressively adjusting the amount of spin on each wheel. Starting with a sharper turn and greater spinning on one side of the wheel, the amount of spinning on both sides finally approaches parity. The robot will proceed straight till a spiral can be constructed unless it senses an item or there is not enough room. On the route, the robot may take arbitrary detours at any time. See Figure 2.11.

Figure 2.11

Spiral Path Algorithm Simulation

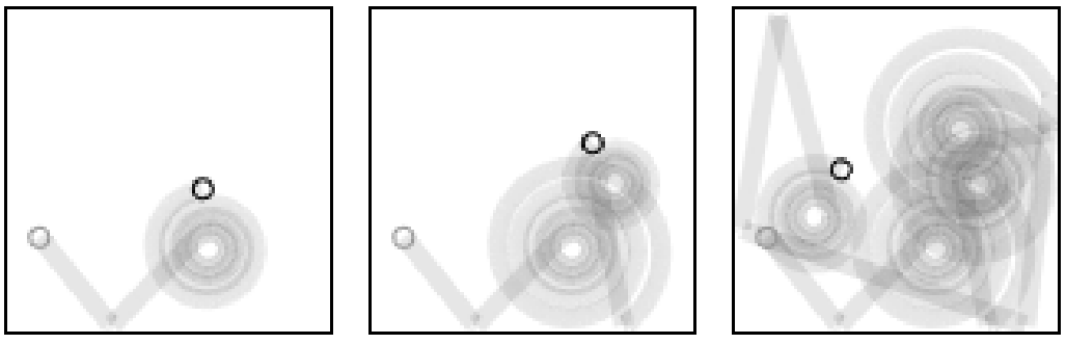
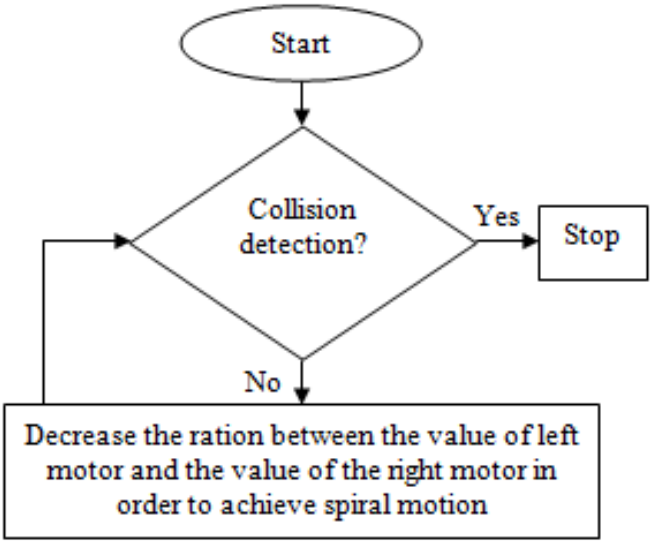


Figure 2.12

Flow Chart of The Spiral Algorithm. (Hasan, 2014)



3. Snaking or 'S' Shaped Pathway Algorithm

For domestic robots, path planning based on a snake-like movement is also typical. For a demonstration of movement in a room, see Figure 2-13. Realistically, this pattern produces a lot of errors because it may need to stop and turn a lot. Consider the misaligned wheels or the cheap and inaccurate sensors. Errors will accrue over time.

In order to avoid positional errors accumulating, the authors advise employing the snaking pattern only for local, small areas that are removed fast. Combining a random walk path with a snaking path will allow the random walk to explore the room while

the snaking pattern covers the actual area. They claim that doing this is more effective than using the random or winding path independently.

Figure 2.13

Snaking Pathway Algorithm Simulation

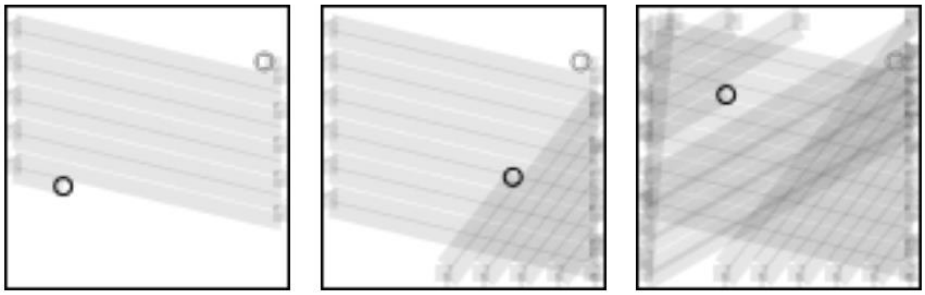
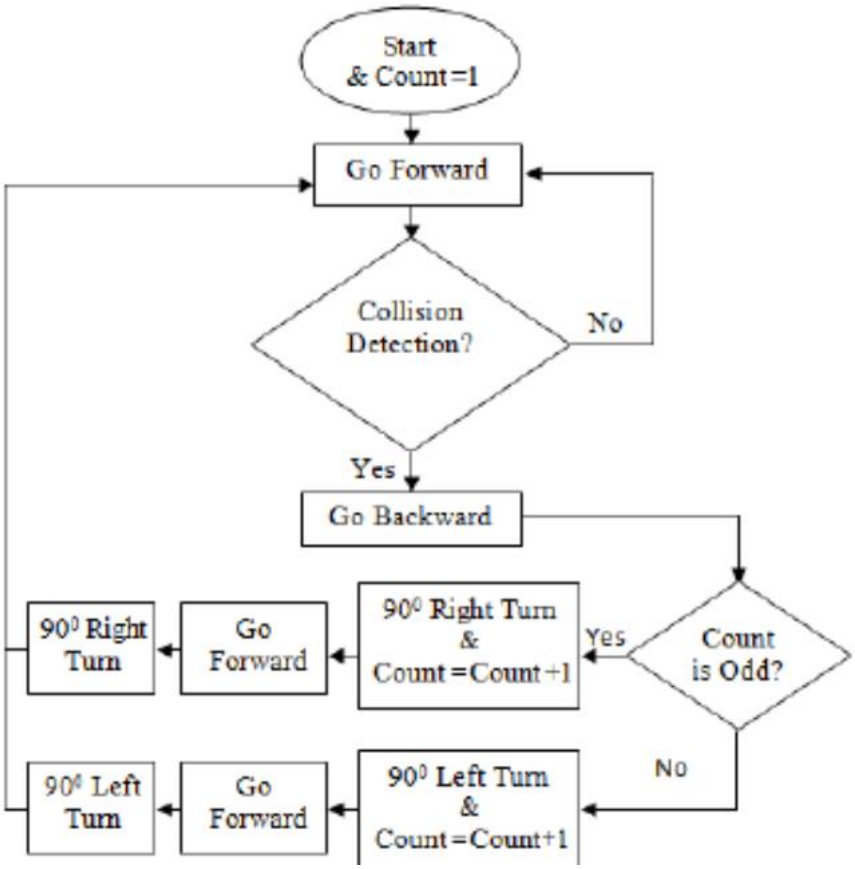


Figure 2.14

Flow Chart of The Snaking Pathway Algorithm (Hasan, 2014)



4. Wall Following Algorithm

The utilization of wall follow pathways primarily serves to facilitate the robot in familiarizing itself with the boundaries of a room. The robot progresses according to the instructions provided by the developers until it encounters a wall. Subsequently, the robot executes rotations to circumvent collisions, enabling it to navigate along the edge of the obstacle. Wall following proves particularly advantageous in effectively clearing challenging areas, such as corners. To get adequate coverage, the authors recommend combining it with other strategies. See the demonstration in Figure 2.15.

Figure 2.15

Wall Following Algorithm Simulation

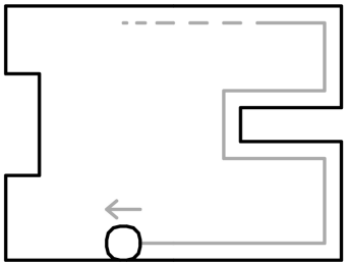
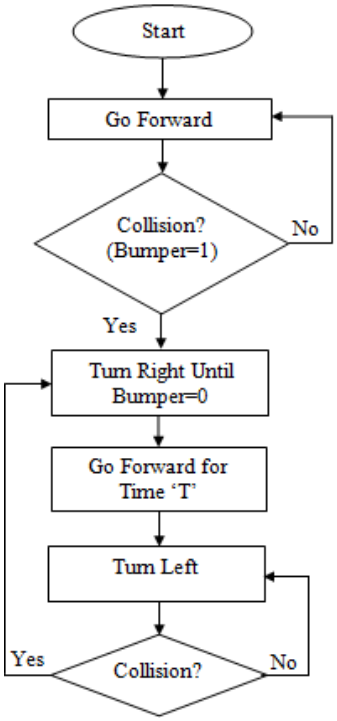


Figure 2.16

Flow Chart of Wall Following Algorithm. (Hasan, 2014)



2.3.2 Path Transforms (Sörme, 2018)

The authors suggest a method called path transform. This solution is designed to achieve complete coverage from an initial starting point to a designated destination point within a known environment. Picture the environment as a grid composed of square cells. Each cell is assigned values based on factors such as the distance to the goal, the proximity to barriers, and the level of obstruction or discomfort.

This strategy aims to have a robot follow the shape patterns of obstacles in the environment in addition to the circular contour patterns that radiate from the objective point. resulting in a final product with fewer turns in this way. When utilizing this technique, a robot needs to be extremely accurate and cannot only rely on dead reckoning. We require a sensor. The landmarks that are seen by the sensor are compared to a previously computed map.

2.3.3 Reactive Approaches (Patle, 2019)

1. **Genetic Algorithm (GA)**

In this approach, the problem is addressed by assigning a population—comprising individuals with unique genetic traits—to the given problem. Each member of the population is then evaluated based on an objective function, yielding a fitness value. Individuals with higher fitness values are selected to contribute to the next generation through crossover, allowing them to pass on their genetic traits. Mutation is introduced to maintain population diversity and prevent premature convergence. The algorithm terminates once the population converges. Despite its randomized nature, Genetic Algorithms (GAs) outperform random local searches by leveraging historical data in addition to its ability to explore solution space systematically.

2. **Fuzzy Logic (FL)**

Genetic Algorithms (GAs) find application in scenarios characterized by high complexity, nonlinearity, and unpredictability. These applications encompass data classification, autonomous control, pattern recognition, and decision-making tasks. GAs excel in transforming human-supplied rules (If-Then statements) into their mathematical equivalents, leveraging the inherent flexibility and adaptability of evolutionary principles to address diverse problem domains.

3. Neural Network (NN)

An Artificial Neural Network (ANN) constitutes an intelligent system comprised of numerous interconnected processing components. These components, also known as neurons, possess the ability to dynamically respond to external inputs by transmitting information. The ANN is typically represented by layers of interconnected nodes, organized in a structured manner to effectively process and transmit information. There is an activation function in the nodes. Through a system of weighted connections, these patterns subsequently communicate with buried layers for actual processing. The output layer connects with the hidden layers to provide the needed response. Mobile robot navigation benefits from NN's generalization, huge parallelism, distributed representation, learning, and fault tolerance properties.

4. Firefly Algorithm (FA)

The fundamental concept behind the Firefly Algorithm is inspired by the stochastic survival behavior of fireflies in nature, which emit light as a means of communication. Fireflies, also known as lightning bugs, employ this light emission to attract mates, convey messages, and sometimes deter predators.

5. Particle Swarm Optimization (PSO)

The Particle Swarm Optimization (PSO) algorithm emulates social animal behavior, yet it operates without a designated group leader. Consisting of a collection of particles, each representing a potential solution, the PSO algorithm harnesses the collective intelligence of these particles to explore and optimize problem spaces.

6. Ant Colony Optimization (ACO)

The method utilized to resolve the combinatorial optimization problem is population-based. Ants' ability to locate the quickest route from their nest to a food source is where the ACO algorithm got its start.

7. Bacterial Foraging Optimization (BFO) Algorithm

Chemotaxis, a central aspect of the Bacterial Foraging Optimization (BFO) algorithm, involves bacteria detecting chemical gradients and exchanging specialized signals with

one another. Chemotaxis, swarming, reproduction and eradication, and dispersal constitute the four core principles of the BFO algorithm. Here's how bacteria behave when hunting for nutrients:

- Bacteria continually move across the environment in search of nutrient-rich areas. Those in regions with less food disperse and eventually perish, while bacteria in nutrient-abundant areas thrive and reproduce by dividing into two equal halves.
- Chemical signals attract bacteria from nutrient-rich areas towards those with lower nutrient concentrations, and vice versa. Bacteria in areas with lower nutrients emit specialized signals to deter those in nutrient-rich areas.
- Highly nutrient-rich areas are mapped out by bacteria through their movement and interactions.
- Once a nutrient-rich area is identified, bacteria spread out again in search of new sources of nutrition, contributing to the ongoing exploration and exploitation of resources in their environment.

8. Artificial Bee Colony (ABC) Algorithm

The Artificial Bee Colony (ABC) algorithm comprises a population of candidate solutions, analogous to a source of food for bees. This algorithm, belonging to the domain of swarm algorithms, is characterized by its population-based stochastic search strategy, known for its simplicity and computational efficiency. The ABC's food search cycle is governed by three key rules:

- Employed bees are dispatched to various food sources to evaluate the quality of the nectar.
- Food sources are selected based on the assessments conducted by employed bees, considering the quality of the nectar.
- Scout bees are tasked with identifying potential food sources and guiding other bees towards them for further evaluation.

9. Cuckoo Search (CS) Algorithm

The Cuckoo Search (CS) algorithm is inspired by the behavior of certain cuckoo species, which lay their eggs in the nests of other host birds. This algorithm operates based on three fundamental principles for optimization problems:

- Each cuckoo lays one egg at a time in a randomly selected nest.
- The best nests and eggs are retained and passed down to subsequent generations.
- The number of potential host nests remains constant, and the probability of a host bird discovering a cuckoo egg is determined by the parameter pa (ranging between 0 and 1). In response, the host bird can either remove the egg or abandon the nest and construct a new one.

One application where performance and computing time are critical is mobile robot navigation. The CS algorithm offers an enhanced solution in this context by accelerating convergence rates and improving efficiency.

10. Shuffled Frog Leaping Algorithm (SFLA)

In the realm of engineering optimization, the Shuffled Frog-Leaping Algorithm (SFLA) has emerged as a popular choice. It distinguishes itself from other metaheuristic algorithms due to its superior convergence speed, simplicity of implementation, minimal parameter requirements, higher success rates, and enhanced search capabilities, particularly in uncertain environments. Today, SFLA is extensively utilized for addressing various engineering optimization challenges, with mobile robot navigation serving as a prominent example.

11. Other Miscellaneous Algorithms (OMA)

2.4 Related Work and Research

1. **Stetro -Deep Learning Powered Staircase Cleaning and Maintenance Reconfigurable Robot (Ramalingam, Et Al., 2021)**

Figure 2.17

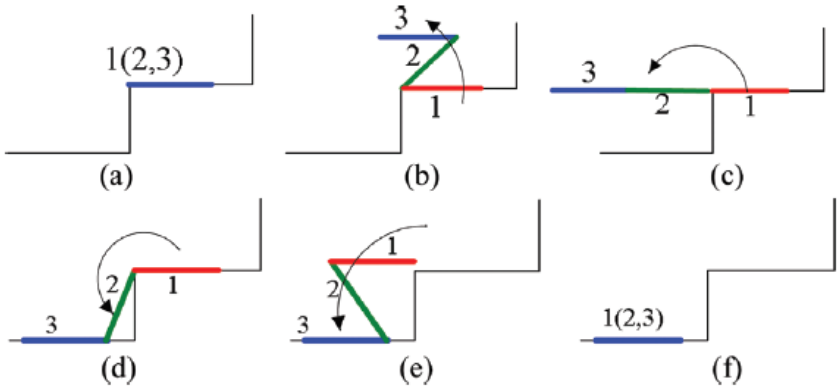
sTetro

The paper proposes an operational framework for the modular and reconfigurable sTetro maintenance robot, leveraging the vision-based EPS (Environment Perception System). The suggested system leverages a real-time object identification model from SSD MobileNet to identify debris, obstructions, and staircases. Additionally, the model eliminates erroneous staircase detection by combining depth information with a MobileNet and SVM. The first step of the staircase is localized by the system using a contour detection technique, while obstacles and debris are located using a depth clustering strategy. The framework has been tested with multistory staircases while being installed on the sTetro robot utilizing NVIDIA's Jetson Nano hardware.

2. **A New Compact Stair-Cleaning Robot (Zhang, 2016)**

Figure 2.18

Z-Shape Legs Robot

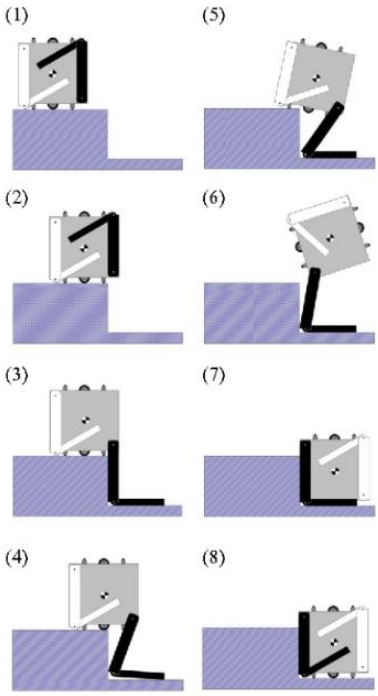


Due to the utilization of only one motor for powering the climbing mechanism and the ability to retract the legs at both ends of the robot when not climbing, the new robot exhibits a significantly more compact design compared to most conventional stair-cleaning or climbing robots. Its compact structure provides a notable advantage, allowing it to navigate along the riser for cleaning stairs effectively. The cleaning tool follows a layout similar to modern ground cleaning robots, while the sensing and control system is engineered to ensure safe stair climbing and prevent collisions and falls. Studies demonstrate that the new robot can successfully ascend or descend various types of stairs, enabling it to carry out cleaning tasks effectively on different staircases.

3. Study on Mobile Mechanism for A Stair Cleaning Robot (T. Kakudou, 2011)

Figure 2.19

L-Shaped Legs Robot



The proposed cleaning robot features a rectangular solid body with L-shaped legs on each side. Through the rotation of its torso, the robot positions its L-shaped legs to stand on its top and bottom sides, facilitating the descent of stairs. This work outlines the mobile mechanism and control strategy employed for descending stairs and translating between staircases.

4. **Ascender (Liszewski, 2023)**

Figure 2.20

Ascender

Ascender, the name of this robot, was just launched on June 2023. In the trailer, it is demonstrated that the robot can move to clean stairs from left to right while swinging its left and right body to the next step. However, they make no mention of the item's ability to descend stairs on their website.

2.5 Comparing Research Table

Name	Design	Sensor	Can Ascend	Can Descend	Height Of Robot (cm)	Width Of Robot (cm)
stetro	vertical conveyor belt	Camera	Yes	Yes	43	42.9
Z-Shape Legs Robot	Z-shape leg	Rear proximity sensor and Contact Board and Ultrasonic	Yes	Yes	15.1	23
L-Shaped Legs Robot	L-shape leg	PSD SENSORS	No	Yes	46	24
This thesis	Scissor Lift	Ultrasonic and switch	Yes	Yes	11.1	38

CHAPTER 3
METHODOLOGY

The following sections detail the techniques used to implement the robot's ability to climb and descend stairs.

3.1 Overview

This thesis presents the development of a stair-climbing robot featuring a scissor lift mechanism controlled by stepper motors, integrating ultrasonic sensors and microswitches for precise stair detection. Utilizing an Arduino Mega microcontroller as the central control unit, the robot's components are predominantly fabricated via 3D printing technology, enabling customized designs and rapid prototyping iterations. The scissor lift mechanism, driven by stepper motors, enables controlled vertical movement, facilitating both ascent and descent on staircases of varying heights and angles.

3.2 Mechanical Design

The system's body comprises three main components: a top platform housing the stepper motor, a bottom platform accommodating mecanum wheels, and a middle section serving as the linkage for the scissor lift mechanism, connecting the top and bottom platforms.

Figure 3.1

Isometric View of 1 Leg

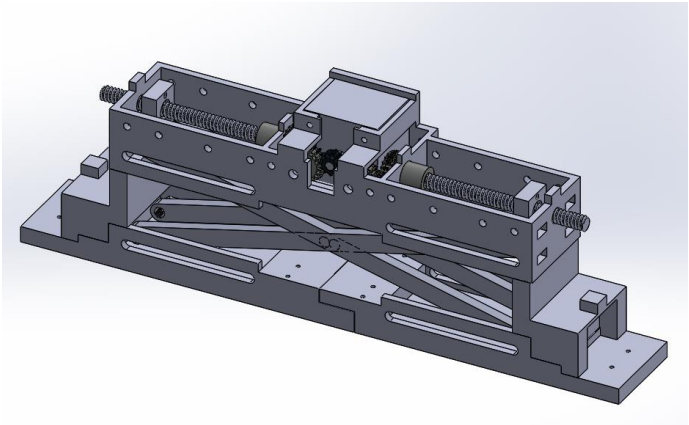
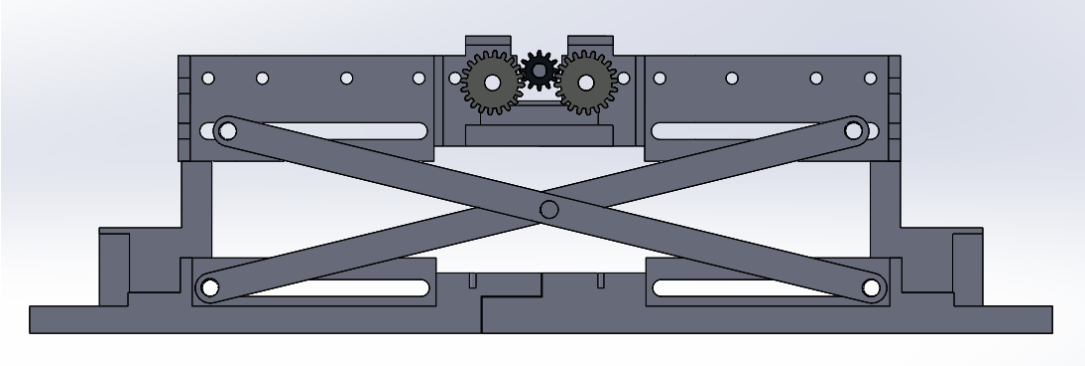


Figure 3.2

Front View of 1 Leg



The stepper motor is employed to drive gears, with separate left and right gears rotating the shaft connected to the linkage, enabling vertical movement of the robot. By utilizing this mechanism, when the middle section rotates clockwise, the left gear ascends while the right gear descends, allowing for differential movement. This configuration facilitates precise control over the shaft's position, enabling it to transition from the outermost edge of the platform to the center for both the left and right sides. The system incorporates a gear ratio to augment torque, achieved by utilizing a smaller drive gear with 12 teeth alongside a larger gear with 20 teeth. The system integrates two bevel gears, each with 20 teeth, to collectively drive both the front and side gears.

Figure 3.3

Two Bevel Gears

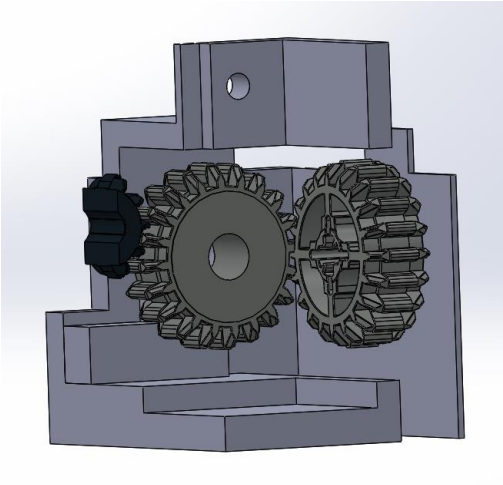
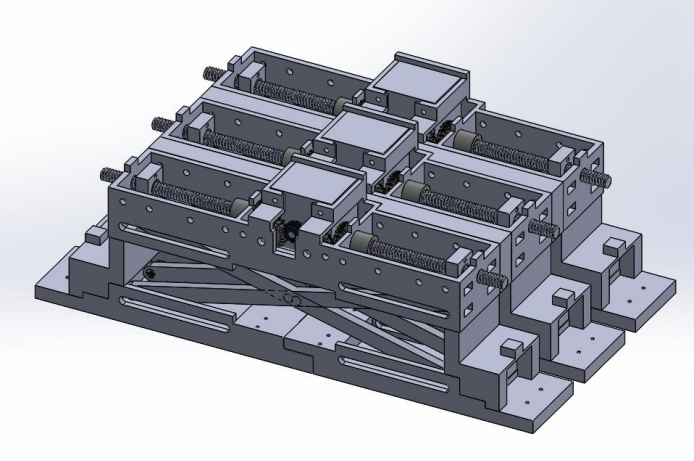


Figure 3.4

Isometric View of All 3 Legs



3.3 Equipment Selection

1. Mecanum Wheel

Figure 3.5

Mecanum Wheel

Table 3.1

Mecanum Wheel's Specification

Parameter	Description
Diameter	60 mm.
Body material	ABS Plastic
Weight	46 g
Load capacity	5 kg
Number of rollers	9

2. **Stepper Motor**

Figure 3.6

Stepper Motor

Table 3.2

N20 Gear Motor with Encoder's Specification

Parameter	Description
Voltage	4.83V
Rated current	1.5 A
Max speed	1-1000 rpm
Torque Max	0.4 Nm
Length	34 mm

3. **N20 Gear Motor with Encoder**

Figure 3.7

N20 Gear Motor with Encoder

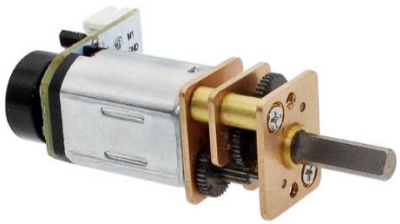


Table 3.3

N20 Gear Motor with Encoder's Specification

Parameter	Description
Voltage	12V
Gear ratio	1:150
Pluse per revolution	1050
Weight	13 g

4. L298N

Figure 3.8

L298N

Table 3.4

L298N's Specification

Parameter	Description
Logic Voltage	5 V
Driver Voltage	5-35 V
Driver Current	2 A
Logical Current	0 – 36 mA
Maximum Power	25 W

5. Ultrasonic

Figure 3.9

Ultrasonic

Table 3.5

Ultrasonic's Specification

Parameter	Description
Sensing Range Min	20 mm
Sensing Range Max	4.5 m
Beam Angle	15°

6. Micro Switch

Figure 3.10

Micro Switch

Table 3.6

Micro Switch's Specification

Parameter	Description
Actuator	Roller Lever
Current Rating	600 mA
Operating Force	1.96 N

7. Arduino Mega 2560

Figure 3.11

Arduino Mega 2560

Table 3.7

Arduino Mega 2560's Specification

Parameter	Description
Microcontroller	ATmega2560
Operating Voltage	5 V
Input Voltage	6 – 20 V
Digital I/O Pins	54 (of which 15 provide PWM output)
DC Current per I/O Pin	20 mA

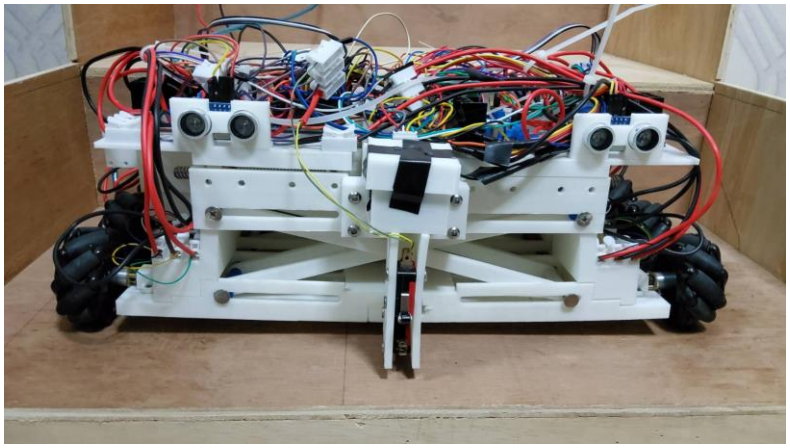
3.4 Actual Robot

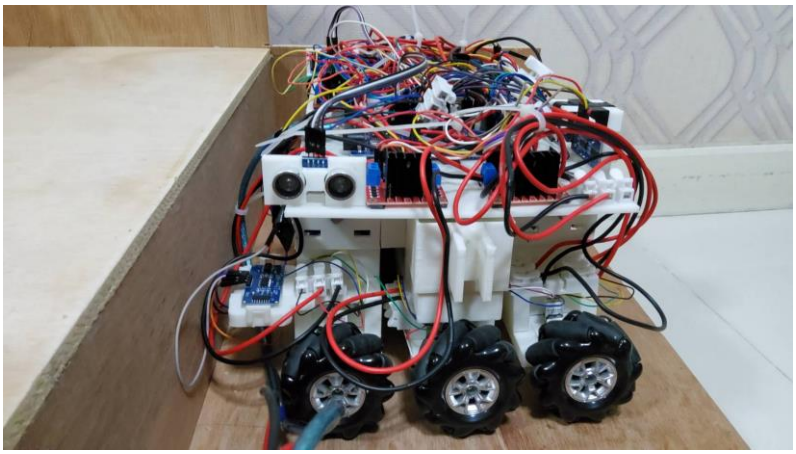
The robot boasts dimensions of 38 cm in width, 11.1 cm in height, and 19.36 cm in length, with a maximum stair step height capability of 13 cm. Its leg configuration showcases three Creality stepper motors, specifically the Z-axis 42-34 model, acclaimed for their robust torque capabilities, boasting a maximum torque rating of 0.4 Nm each. Each stepper motor is meticulously allocated to govern the movement of an individual leg, guaranteeing meticulous and synchronized motion throughout stair traversal maneuvers.

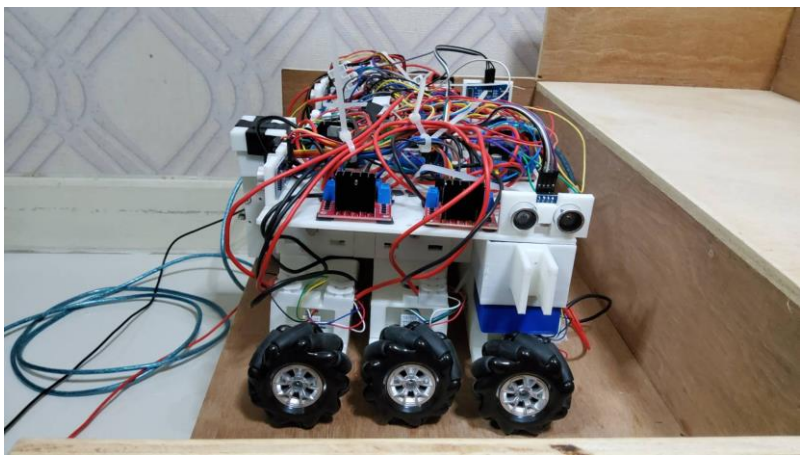
The leg mechanism incorporates a scissor lift design, enabling vertical movement for ascending and descending staircases. This design not only facilitates stair traversal but also ensures that the robot maintains an appropriate height when navigating flat surfaces, allowing for unhindered movement and maneuverability.

Figure 3.12

Actual Robot





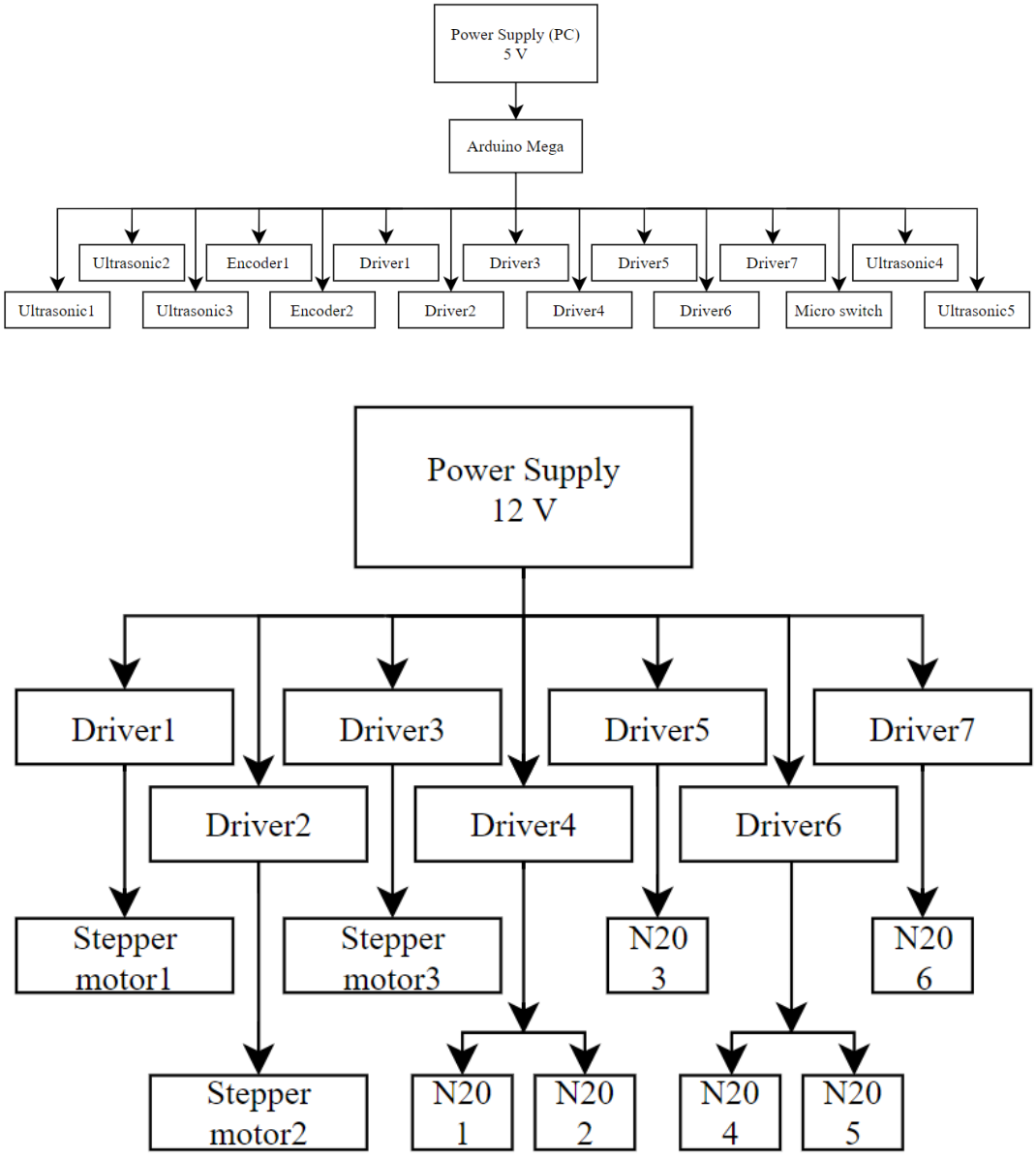


3.5 Block Diagram

1. Block Diagram of The Power Supply System of The Robot

Figure 3.13

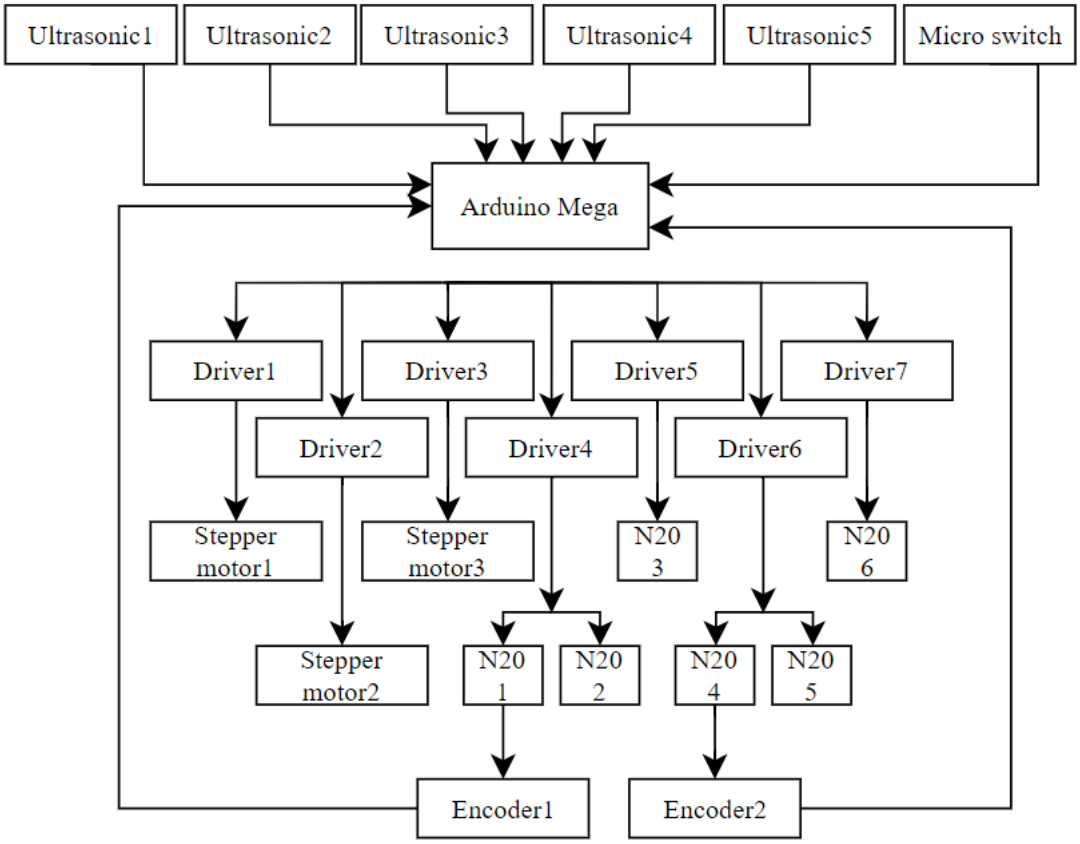
Block Diagram of The Power Supply System



2. Block Diagram of The Electrical System of The Robot

Figure 3.14

Block Diagram of The Electrical System



The primary control system of the robotic platform is centered around an Arduino Mega microcontroller, leveraging ultrasonic sensors for floor and wall detection, as well as micro switches for stair recognition. Floor detection is facilitated by an ultrasonic sensor, which assesses the distance to the floor surface. Should this distance exceed a predefined threshold, indicating the presence of a significant drop such as a stairwell, the robot initiates a descent maneuver.

Conversely, a separate ultrasonic sensor is deployed to gauge the distance to adjacent walls. Should this distance fall below a predetermined threshold, indicative of proximity to a wall, the robot executes a 90-degree turn and advances towards the staircase for ascent.

To further improve the robotic platform's ability to traverse stairs, micro switches located at the front of the robot are strategically placed to detect when the robot has

reached the base of the staircase. This detection signals the robot's readiness to initiate the climbing operation. Propulsion is achieved through the implementation of N20 gear motors, offering robust and precise movement control.

For the intricate task of ascending and descending staircases, the robot relies on three stepper motors, each dedicated to controlling a separate leg. This arrangement ensures stability and control during stair navigation.

Furthermore, wheel movement is monitored via encoders, providing real-time feedback on wheel rotation and enabling precise navigation and motion planning. Collectively, these components form a sophisticated and efficient system, enabling the robotic platform to autonomously traverse staircases with accuracy and reliability.

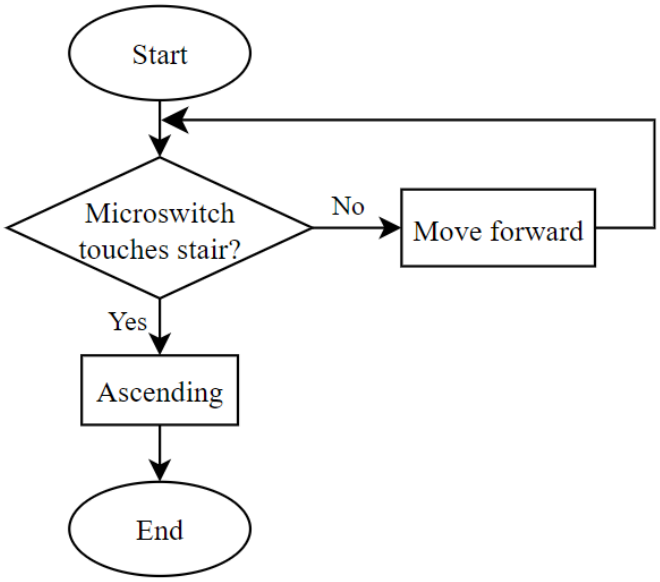
3.6 Control on The Preparation for Ascending and Descending

1. **Algorithm1:** Front Facing, Stair Climbing Up.

This algorithm is straightforward and relies on placing the robot close to the stairs, facing the stair raiser, and using a front micro switch to detect collisions. It's a simple approach suitable for basic stair climbing tasks. In the algorithm, C_F signifies the collision state detected by the front microswitch. When a collision is detected, C_F is set to true; otherwise, it is set to false.

Figure 3.15

Flowchart of Algorithm1

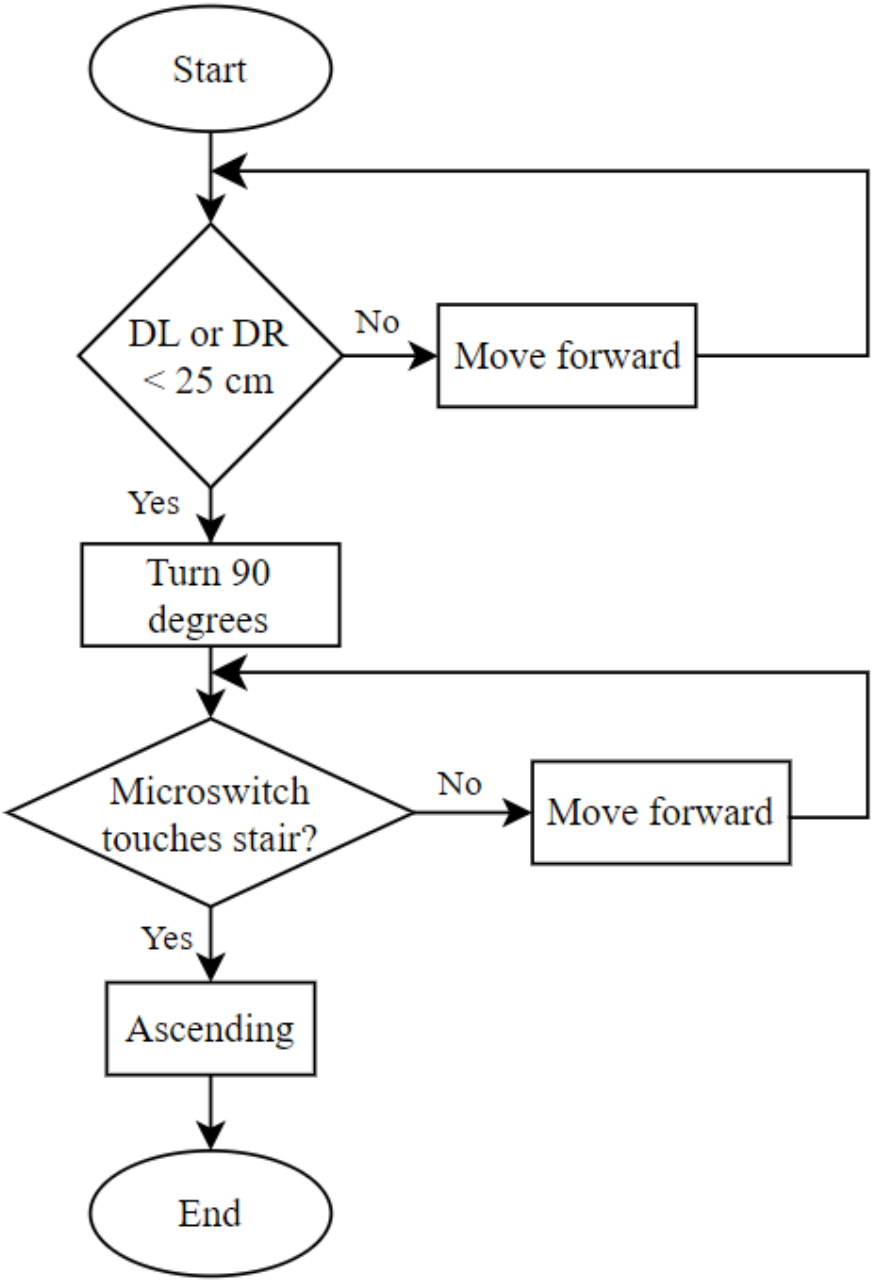


2. **Algorithm2:** Side Facing, Stair Climbing Up

I'm positioning the robot perpendicular to the stair and using ultrasonic sensors on the robot's side to detect when the robot reaches the stair. Once detected, then switch to Algorithm1 to continue moving up the stairs. In the algorithm, D represents the distance detected by the ultrasonic sensor. D_l indicating distance detected by the left ultrasonic sensor, D_r denoting distance detected by the right ultrasonic sensor.

Figure 3.16

Flowchart of Algorithm2

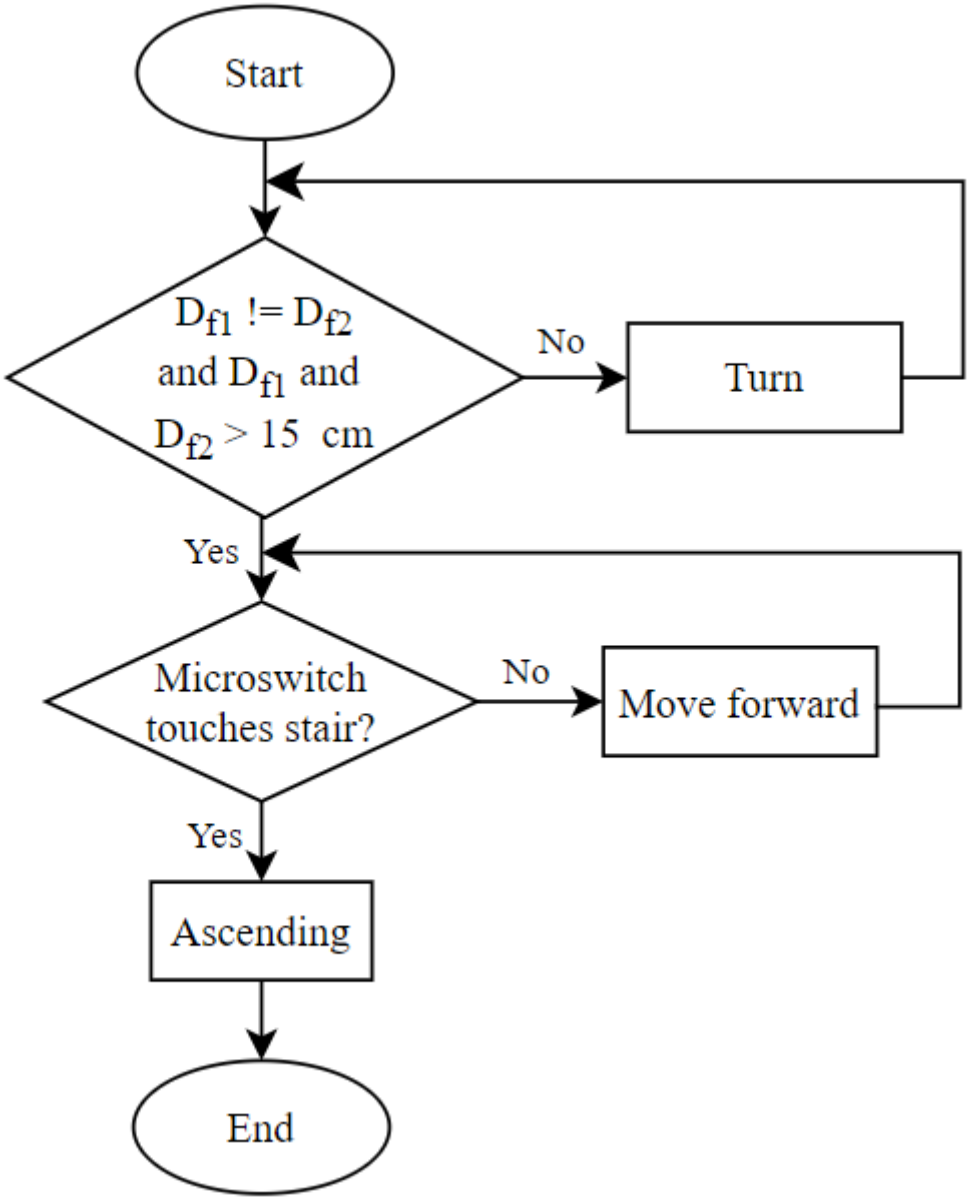


3. **Algorithm3:** Non-Perpendicular, Stair Climbing Up.

To achieve this, the robot undergoes a turning process until its two front ultrasonic sensors detect nearly identical ranges, both measuring less than 15 cm. Once this condition is met, the robot halts its movement, initiating Algorithm 1 to proceed with ascending the stairs. Here, D_{f1} represents the distance detected by the first front ultrasonic sensor, while D_{f2} denotes the distance detected by the second front ultrasonic sensor.

Figure 3.17

Flowchart of Algorithm3

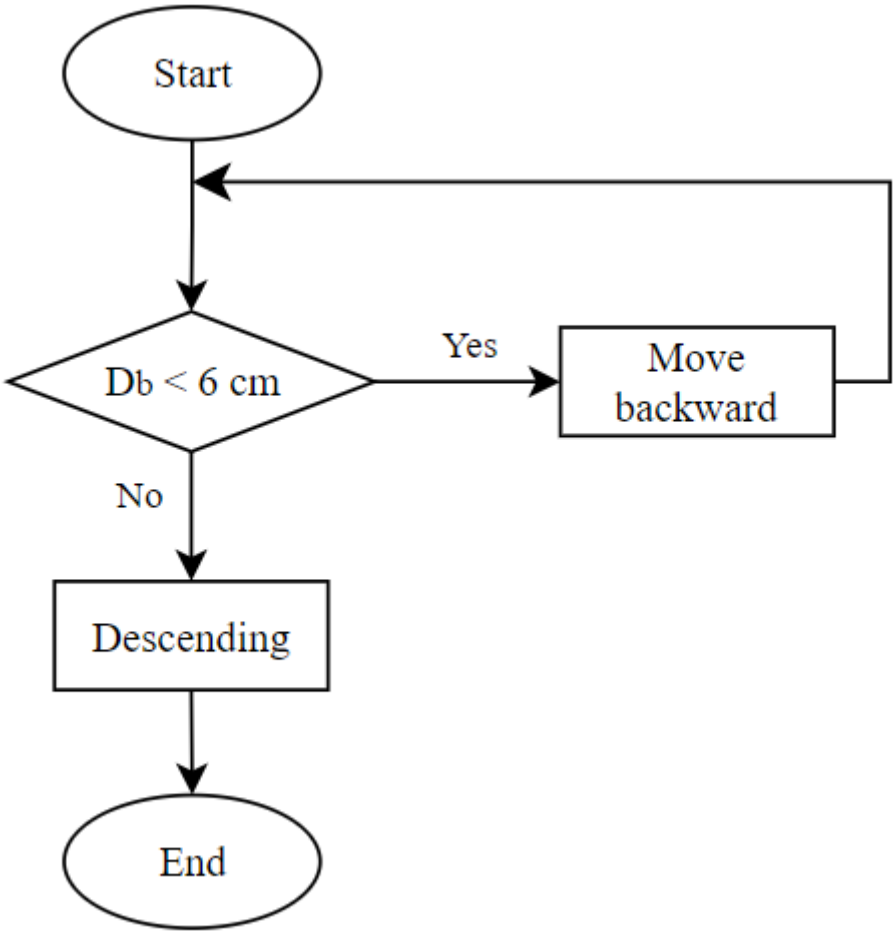


4. **Algorithm4:** Descending Stairs

I placed the robot on the step and used ultrasonic sensors to detect the floor below. This approach allows the robot to gauge the distance to the next step or floor level, enabling controlled descent. D_b representing distance detected by the back ultrasonic sensor.

Figure 3.18

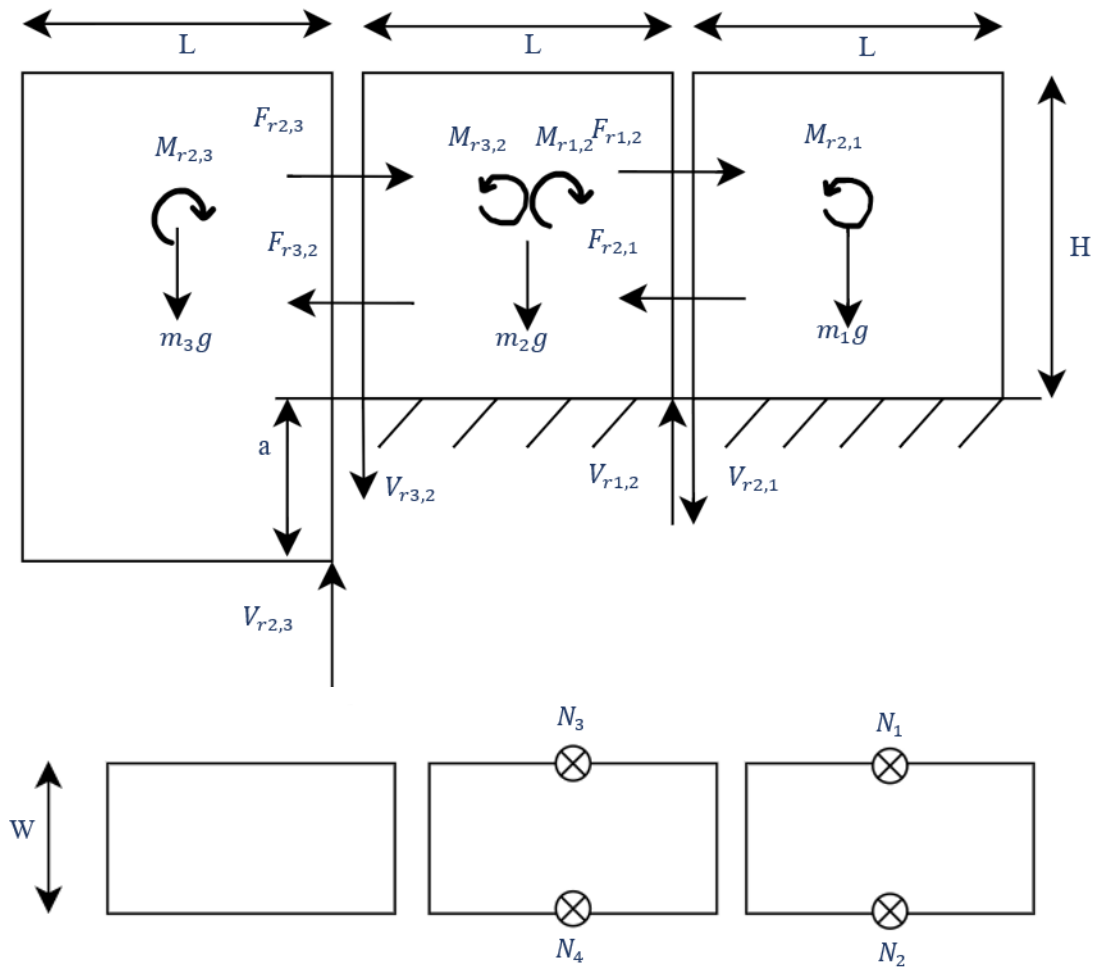
Flowchart of Algorithm4



3.7 Centre of Gravity (CG) Analysis

Figure 3.19

Free Body Diagram



$$\sum \vec{F} = 0$$

$$\therefore m_1g + m_2g + m_3g = N_1 + N_2 + N_3 + N_4$$

$$\sum \vec{M} = 0$$

For block 1

$$\sum \vec{F}_x = 0$$

$$\therefore F_{r1,2} = F_{r2,1}$$

$$\sum \vec{F}_y = 0$$

$$\therefore m_1 g + V_{r1,2} = N_1 + N_2$$

$$\sum \vec{M} = 0$$

$$\therefore \frac{m_1 g L}{2} - \frac{(N_1 + N_2)L}{2} = M_{r2,1}$$

For block 2

$$\sum \vec{F}_x = 0$$

$$\therefore F_{r1,2} + F_{r2,3} = F_{r2,1} + F_{r3,2}$$

$$\sum \vec{F}_y = 0$$

$$\therefore m_2 g - V_{r1,2} + V_{r3,2} = N_3 + N_4, V_{r1,2} = V_{r2,1}$$

$$\sum \vec{M} = 0$$

$$\therefore \frac{m_2 g L}{2} - \frac{(N_3 + N_4)L}{2} = M_{r3,2} - M_{r2,1}$$

For block 3

$$\sum \vec{F}_x = 0$$

$$\therefore F_{r2,3} = F_{r3,2}$$

$$\sum \vec{F}_y = 0$$

$$\therefore m_3 g = V_{r3,2}$$

$$\sum \vec{M} = 0$$

$$\therefore \frac{m_3 gL}{2} = M_{r3,2}$$

$$M_{r2,3} = M_{r3,2}$$

For block 1 and 2 as a whole

$$\sum \vec{F}_x = 0$$

$$\therefore F_{r2,3} = F_{r3,2}$$

$$\sum \vec{F}_y = 0$$

$$\therefore (m_1 + m_2)g + V_{r3,2} = N_1 + N_2 + N_3 + N_4$$

$$\sum \vec{M} = 0$$

$$\therefore \frac{3m_1 gL}{2} + \frac{m_2 gL}{2} = M_{r3,2} + \frac{3(N_1 + N_2)L}{2} + \frac{(N_3 + N_4)L}{2}$$

$$\frac{3m_1 gL}{2} + \frac{m_2 gL}{2} = \frac{m_3 gL}{2} + \frac{3(N_1 + N_2)L}{2} + \frac{(N_3 + N_4)L}{2}$$

$$N_3 + N_4 = m_2 g + 2m_3 g$$

$$N_1 + N_2 = m_1 g - 2m_3 g$$

Based on the specified constraints and the extreme case analysis provided, it is evident that the stability of the robot is ensured. The constraints dictate that the normal forces (N_i) exerted by each leg (i) must be greater than or equal to zero. In the extreme scenario where all normal forces are equal to zero, the condition $\therefore m_1 = 2m_3$ is established, ensuring equilibrium.

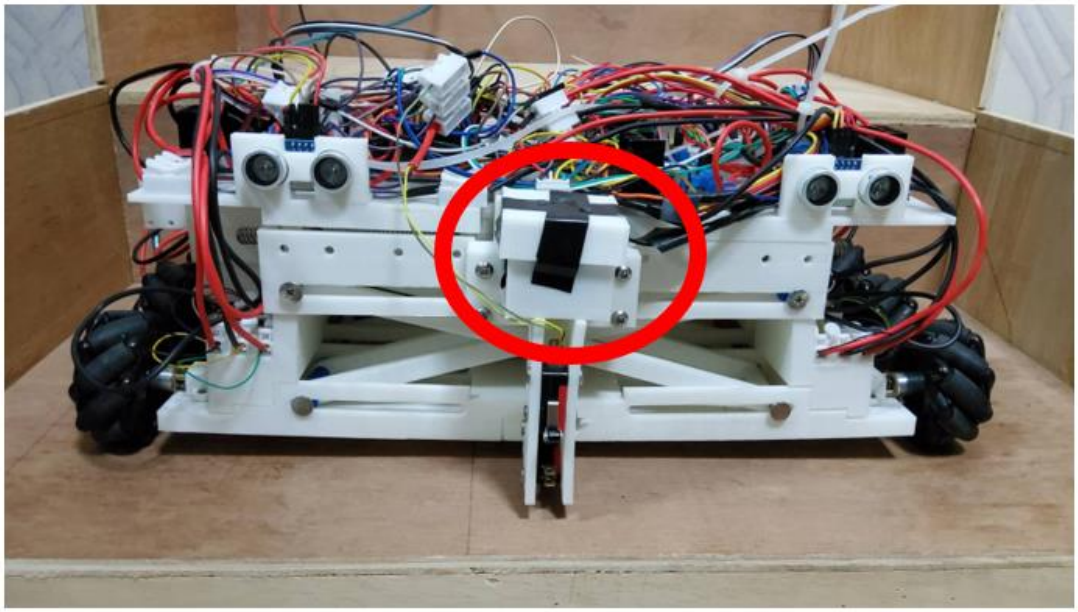
By substituting the dimensions from the design into this condition, it is confirmed that the constraints are satisfied. Consequently, the robot remains in a stable condition, as indicated by the fulfillment of the stability constraints. This analysis underscores the robustness and structural integrity of the robot design, ensuring its stability under various loading conditions and operational scenarios.

To maintain stability and balance during stair climbing, additional weight is strategically positioned at the front of the robot. This counterbalance mechanism ensures that the robot remains steady when the last leg is in motion, providing optimal stability and preventing tipping.

Weight distribution is optimized by strategically adding 195 grams of weight to the foremost section of the robot. This placement ensures a balanced center of gravity, enhancing stability and preventing tipping during stair climbing and other maneuvers.

Figure 3.20

Weight Added in Front



CHAPTER 4
RESULT AND DISCUSSION

4.1 Overview

The Arduino program implemented within the robotic system leveraged the AccelStepper library to intricately control stepper motor operations. This versatile library facilitated precise regulation of position, speed, and acceleration parameters, allowing for tailored adjustments to meet specific application demands. Notably, the ability to set a maximum speed threshold provided the flexibility to optimize efficiency without sacrificing precision. Through meticulous calibration, a speed of 500 steps per second was determined as the optimal velocity for upward movements of the robot, striking a delicate balance between expeditious traversal and resource conservation. This integration exemplified a systematic approach to robotic control, where careful consideration of motor dynamics and operational parameters culminated in enhanced performance and efficiency, underscoring the effectiveness of the implemented system.

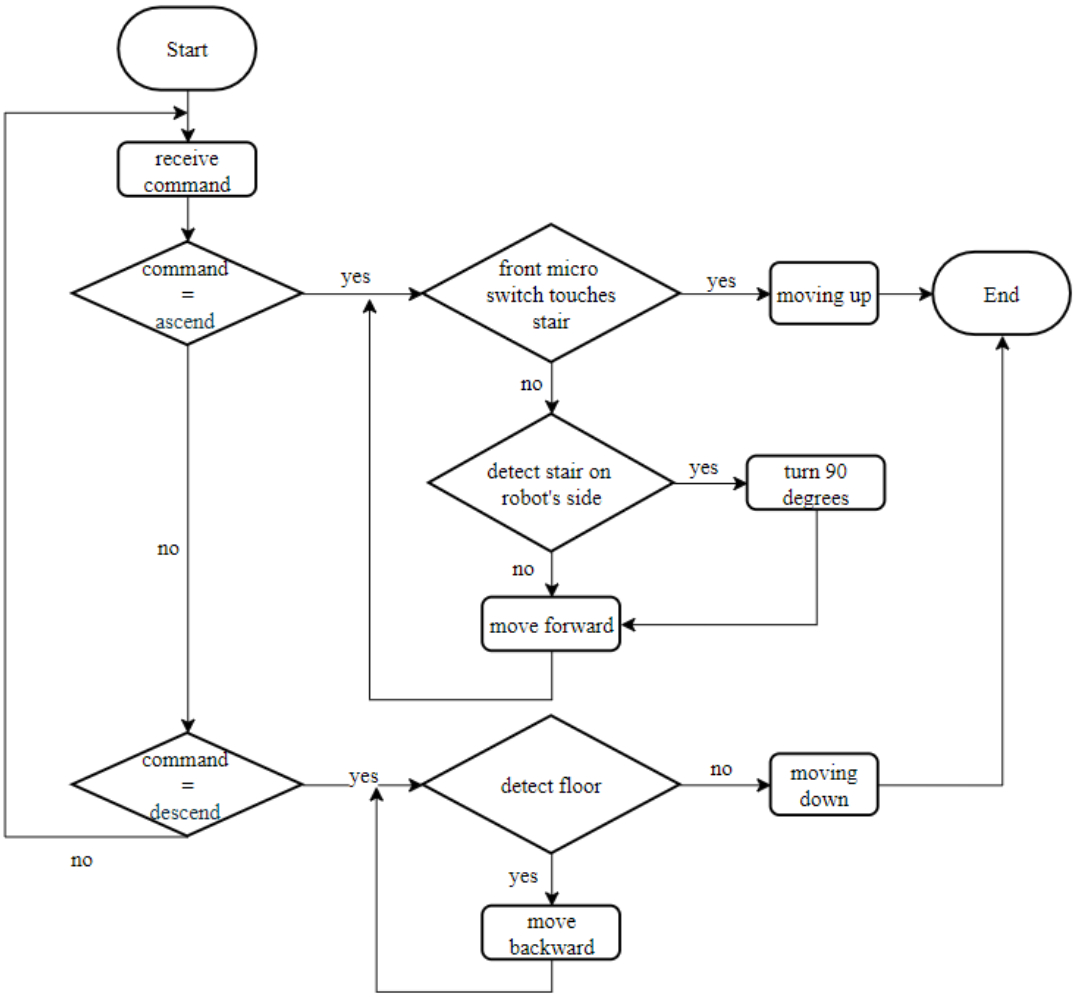
In its dual operational modes, the robot offers distinct functionalities to cater to user preferences. In manual mode, movement commands are directly input via a keyboard interface, while encoder feedback and the AccelStepper library ensure precise control over wheel distance and stepper motor positions. Conversely, automatic mode integrates sensor inputs from ultrasonic, encoder, and micro switch sensors to autonomously navigate the robot, allowing it to ascend or descend staircases by dynamically adjusting its trajectory based on command that it received.

During the ascent process, the robot incorporates an ultrasonic sensor to assess the presence of stairs on its side. Upon detecting stairs, the robot dynamically adjusts its orientation to align its front with the stairwell. Subsequently, it proceeds forward until a micro switch engages with the stair, signaling the commencement of the climbing phase. This procedural integration ensures a methodical approach to stair traversal, wherein the robot seamlessly transitions from lateral scanning to precise alignment and eventual upward movement, thus enhancing its adaptability and operational efficiency in traversing varied terrains. For the descent maneuver, the robot employs its ultrasonic sensor to gauge the proximity to the ground. Once the distance measures below the

predefined threshold of 6 cm, the robot initiates the downward movement from the stairway.

Figure 4.1

Flow Chart of Auto Mode



4.2 Robot Performance

This section delves into the performance evaluation of the robot in manual mode, focusing on the implementation of the encoder library to govern the robot's positional control. In conducting the experiment, the robot was tasked with executing movements of 1 meter in four distinct directions, including forward, backward, left, right, with alterations in heading direction. The outcomes of these movement experiments are meticulously documented in Tables 4.1, while changes in heading direction are delineated in Table 4.2. In each experimental iteration, the designated setpoint was

compared with feedback data retrieved from the robot, along with measurements obtained from the measuring tool. Notably, to ensure robustness, the robot executed each setpoint movement three times, enabling the quantification of error. Furthermore, it's important to note that each setpoint corresponds to a pair of distances (x, y), where the command dictates movement in a specified direction relative to the robot's orientation. For instance, a command to move forward by 1 meter corresponds to the setpoint (1,0), while a leftward movement of 1 meter is represented by the setpoint (0,1). The directional alignment of each setpoint is visually depicted in Figure 4.2, providing a comprehensive overview of the experimental parameters and directional orientations.

Figure 4.2

Movement Coordinate

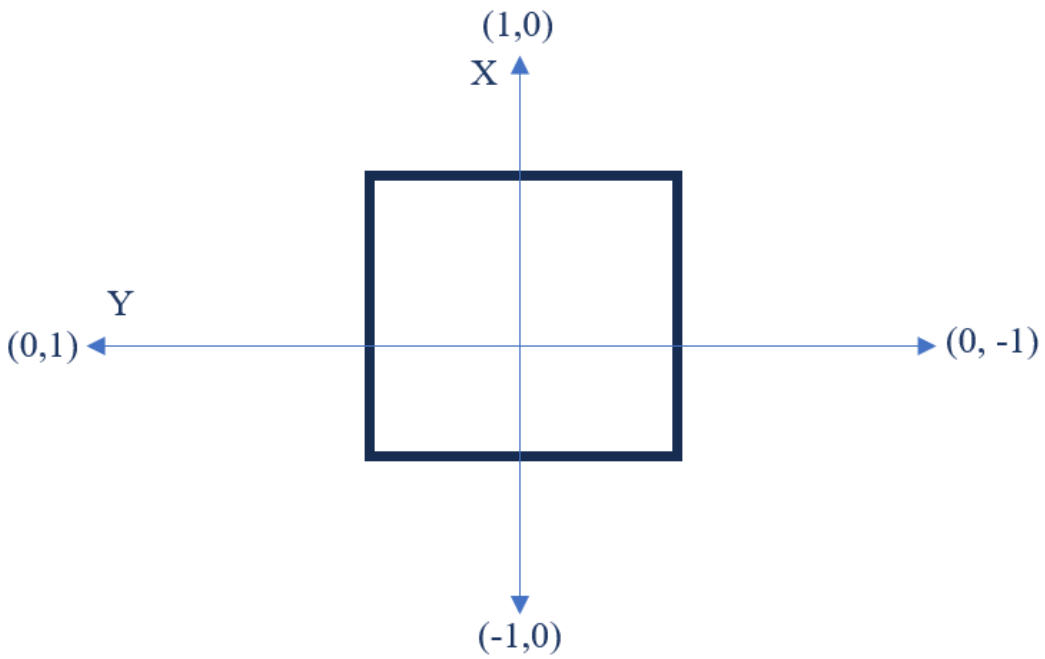


Table 4.1

Movement Experiments

No. of experiment	Coordinate (X, Y)	Measuring	Error
1	(0,0)	(0,0)	0
2	(1,0)	(0.948, -0.024)	0.057
3	(1,0)	(0.97, -0.015)	0.034
4	(1,0)	(0.966, -0.035)	0.049
5	(-1,0)	(-1.014, 0.008)	0.140
6	(-1,0)	(-0.904, 0.08)	0.125
7	(-1,0)	(-0.906, 0.08)	0.123
8	(0,1)	(0.055, 0.749)	0.267
9	(0,1)	(0.075, 0.753)	0.258
10	(0,1)	(0.116, 0.668)	0.352
11	(0,-1)	(0.036, -0.765)	0.238
12	(0,-1)	(0.028, -0.772)	0.230
13	(0,-1)	(0, -0.792)	0.208
Average Error			0.163

Based on the data provided in Table 4-1, all measurements are reported in meters. The error values are derived from a comparison between the final position of the robot and the intended goal location, calculated using the Euclidean distance metric. Analysis of the results reveals a maximum error of 0.352 meters observed for the setpoint (0,1), while the minimum error of 0.016 meters occurs for the setpoint (-1,0). These variations in error can be attributed primarily to the unique design characteristics of the robot, notably its configuration of six mecanum wheels. This configuration introduces increased friction during movement, potentially leading to deviations from the intended trajectory. Furthermore, the non-linear orientation of the wheels' tires, featuring angled profiles, exacerbates this effect by inducing lateral forces that contribute to sliding motion. Consequently, these design factors collectively contribute to the observed discrepancies between intended and actual positions during robot movement trials.

Table 4.2

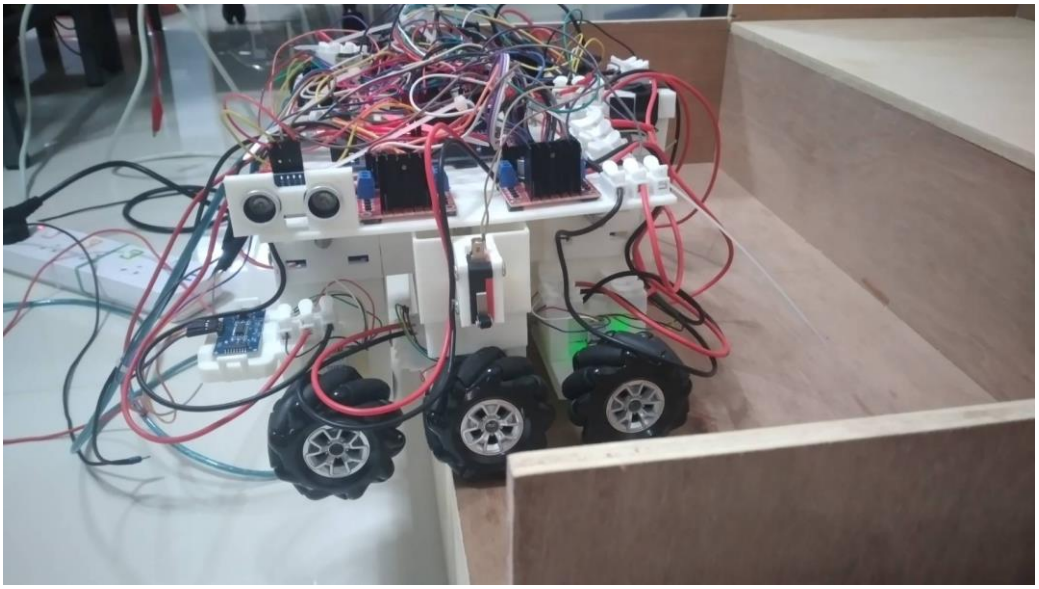
Heading Direction Experiment

No. of experiment	Theta (rad)	Measuring Tool (rad)	Error(Measuring)
1	1.571	1.518	0.053
2	1.571	1.518	0.053
3	1.571	1.536	0.035
4	1.571	1.518	0.053
5	1.571	1.501	0.070
6	-1.571	-1.536	0.035
7	-1.571	-1.553	0.018
8	-1.571	-1.623	0.052
9	-1.571	-1.553	0.018
10	-1.571	-1.501	0.070
Average Error			0.046

Based on the findings presented in Table 4-2, the recorded errors between the goal angle and measured angle exhibit a maximum deviation of 0.07 radians and a minimum deviation of 0.018 radians. Although these errors are relatively modest, it is noteworthy that the robot's positional accuracy may still be affected by subtle discrepancies, particularly attributable to the utilization of six wheels for movement instead of the specified four. This departure from the prescribed wheel configuration introduces a potential source of positional error, particularly concerning the central wheels' influence on the robot's overall alignment. Consequently, while the angular errors remain within acceptable bounds, the deviation in positional accuracy underscores the impact of wheel configuration discrepancies on the robot's performance during movement tasks.

Figure 4.3

Robot Balance when Last Leg in Mid-Air



In Chapter 3, the computation indicates that the mass of leg 1 should be double that of leg 3. Consequently, if additional weight equivalent to the mass of leg 3 is appended to leg 1, the resultant redistribution of mass should confer stability to the robot when leg 3 is suspended in mid-air while negotiating a staircase ascent. This adjustment aligns with the principle of load distribution and equilibrium, wherein the augmented mass on leg 1 compensates for the absence of support from leg 3, thereby enhancing the robot's balance and enabling it to maintain stability during stair traversal maneuvers.

Table 4.3

Ascend and Descend Experiment

No. of experiment	Speed (steps per second)	Time For Ascending (s)	Time For Descending (s)	Time Difference (s)
1	200	47.2	46.09	1.11
2	200	45.73	48.75	3.02
3	200	45.81	45.55	0.26
4	300	31.07	32.06	0.99
5	300	31.11	31.35	0.24
6	300	30.9	31.1	0.2
7	400	25.55	29.64	4.09
8	400	24.56	24.16	0.4
9	400	24.48	26.55	2.07
10	500	19.35	20.11	0.76
11	500	19.92	19.64	0.28
12	500	19.64	20.71	1.07
13	600	16.63	17.77	1.14
14	600	17.52	17.51	0.01
15	600	17.43	18.62	1.19
16	700	16.24	16.05	0.19
17	700	16.04	16.15	0.11
18	700	14.69	17.49	2.8
Average Difference				1.11

Based on the data presented in Table 4-3, the experiments involved ascending and descending the robot using the same speed for the N20 motor controlling the wheels while varying the speed of the stepper motor. It is observed that as the speed of the stepper motor increases from 200 to 700 steps per second, the time required for both ascent and descent decreases. However, it is important to note that there exists a trade-off between speed and torque in stepper motors. While the AccelStepper library allows

for setting a maximum speed of 4000 steps per second, increasing the speed beyond a certain threshold may result in a reduction in torque.

Through experimentation, it is determined that the optimal speed for ascending the robot is 500 steps per second. At this speed, the stepper motor exhibits sufficient torque to effectively move the robot upwards while minimizing the time required for the ascent. This choice strikes a balance between speed and torque, ensuring that the robot can ascend efficiently without sacrificing stability or risking motor stalling. Consequently, utilizing a speed of 500 steps per second for ascending and descending operations optimizes the performance of the robot, enabling swift traversal of staircases while maintaining operational reliability and efficiency.

4.3 Sequence of Climbing Up Stair

Figure 4.4

Sequence of Climbing Up Stair

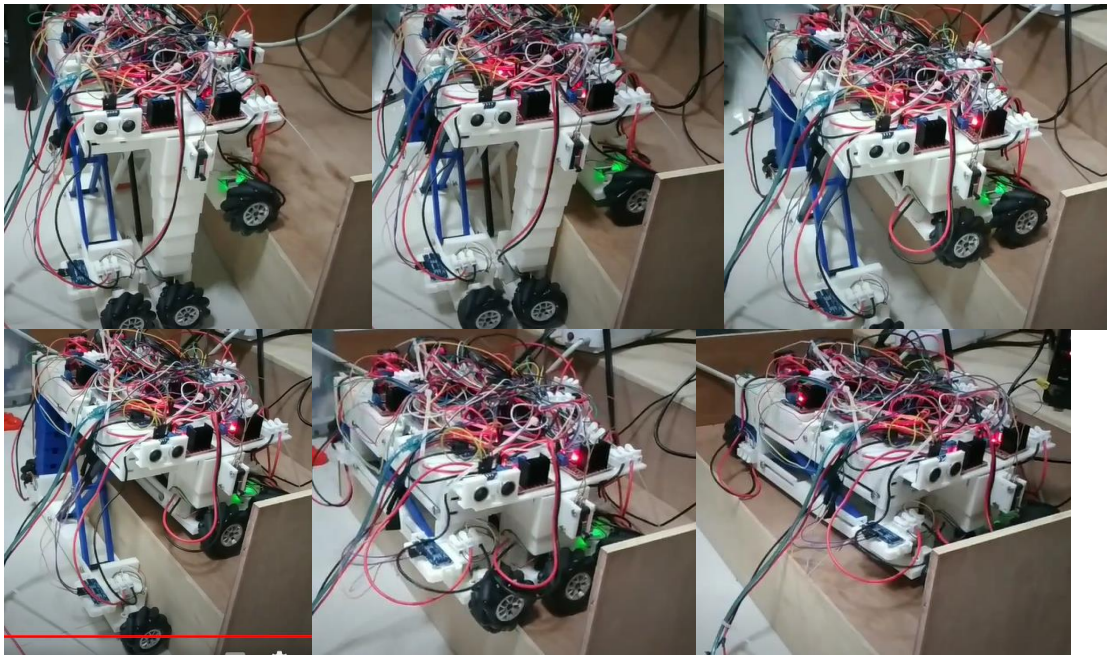


Figure 4.4 illustrates the sequential steps of the robot's movement when it is not aligned with the stairs. Initially, the robot moves forward until the ultrasonic sensor detects a certain distance, prompting a 90-degree turn. Subsequently, it continues forward until a microswitch is activated, signaling it to ascend. The robot then proceeds forward, lifts its first leg, advances again, lifts the second leg, repeats the forward movement, lifts the final leg, and advances once more, completing the sequence.

4.4 Sequence of Climbing Down Stair

Figure 4.5

Sequence of Climbing Down Stair

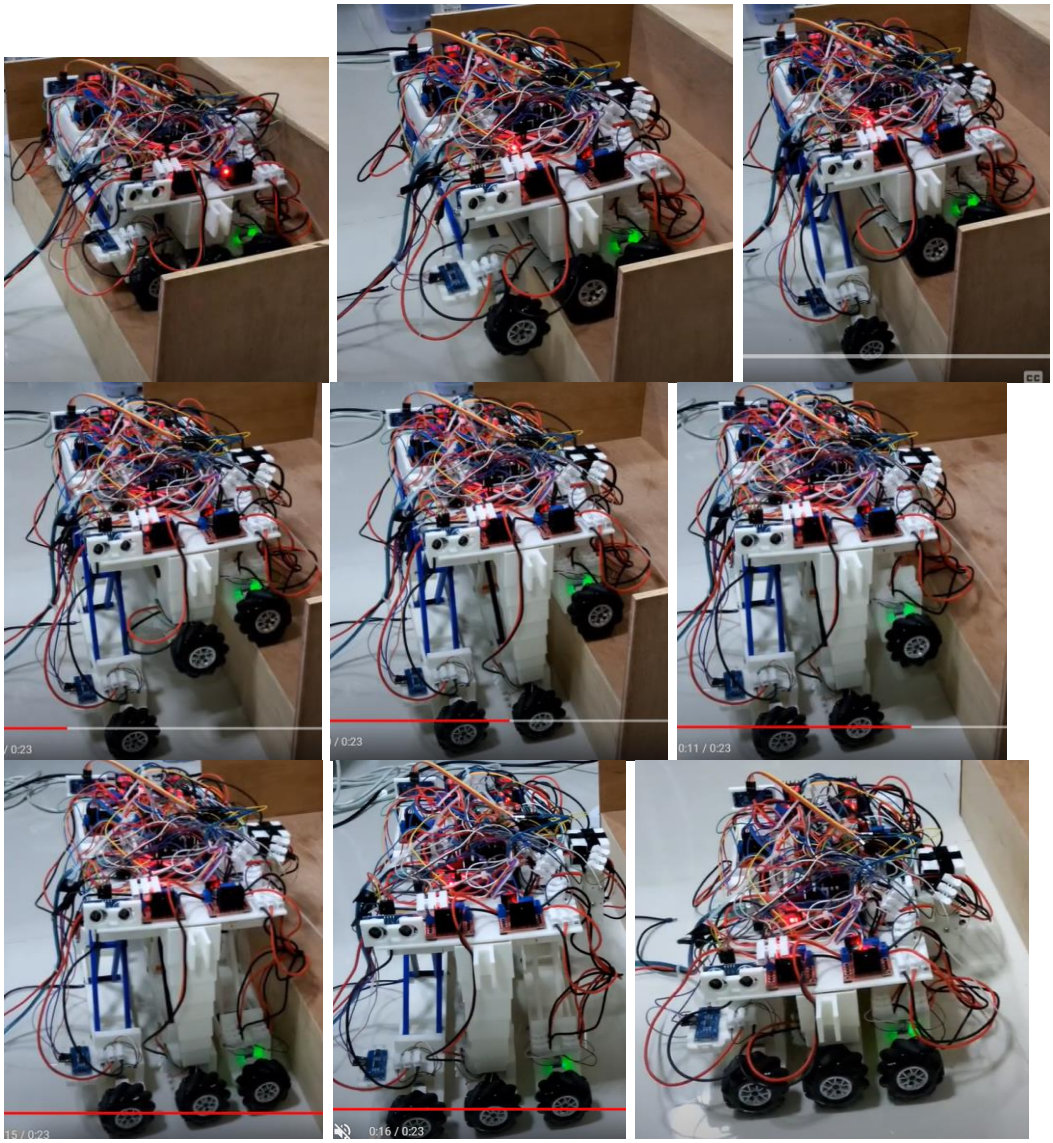


Figure 4.5 depicts the sequential steps of the robot's movement when descending stairs. Initially, the robot moves backward until the ultrasonic sensor detects a specific distance. It then adjusts its position slightly backward before lowering the last leg and moving backward once more. Subsequently, it lowers the second leg, continues moving backward, lowers the first leg, moves backward again, and finally lowers all legs to complete the descent.

CHAPTER 5
RESULT AND DISCUSSION

5.1 Conclusion

In pursuit of designing a preliminary vacuum cleaner robot capable of navigating staircases, this thesis embraced the challenge of optimizing functionality while minimizing costs. Leveraging low-cost sensors and 3D-printed materials, the design epitomized affordability and accessibility in robotics.

Driven by an Arduino microcontroller, interfacing with encoders, ultrasonic sensors, and microswitches facilitated seamless control over N20 gear motors and stepper motors. The incorporation of manual and auto modes underscored the versatility of the system, offering both immediate user control and autonomous operation upon command.

Despite notable achievements, including an average traversal time of 19.9 seconds for staircase navigation, challenges persisted. The mecanum 6-wheel model introduced slight errors in movement, necessitating further refinement. Additionally, the optimization of stepper motor speed at 500 steps per second proved crucial for efficiency.

Furthermore, center of gravity analysis revealed the need for additional front weight to enhance stability, a pivotal insight for future iterations.

In conclusion, while this preliminary design represents a significant step forward, it also serves as a springboard for ongoing innovation. By addressing challenges and refining our approach, we remain committed to realizing an efficient, cost-effective vacuum cleaner robot capable of conquering staircases with precision and reliability.

5.2 Recommendations

1. Adjusting gear ratios, particularly in conjunction with stepper motor, to effectively amplify torque output while simultaneously optimizing speed.

2. To address the limited height capability of the robot, it's recommended to enhance the torque of the stepper motor and consider removing excess support structures to facilitate greater vertical reach.

REFERENCES

B.K. Patle, G. B. (2019). A review: On path planning strategies for navigation of mobile robot,. *Defence Technology*, 582-606.

Dong, P. a. (2016). Design and control of a tracked robot for search and rescue in nuclear power plant. *ICARM*, 330-335.

Eich, M. a. (2009). Adaptive compliance control of a multi-legged stair-climbing robot based on proprioceptive data. *Industrial Robot: An International Journal*, 331-339.

Hartwell, P. (2022, September 14). Retrieved from therobotreport: https://www.therobotreport.com/sensor-breakdown-how-robot-vacuums-navigate-and-clean/

Hasan, K. a. (2014). Path planning algorithm development for autonomous vacuum cleaner robots. *2014 International Conference on Informatics, Electronics and Vision, ICIEV 2014*, 1-6.

Iwan R. Ulrich, F. M.-D. (1997). Autonomous vacuum cleaner. *Robotics and Autonomous Systems*, 233-245.

Jindal, H. &. (2015). Stair Climbing Robot. *International Journal of Scientific Research*.

Jung J, Y. S. (2015). Development of Kinematic 3D Laser Scanning System for Indoor Mapping and As-Built BIM Using Constrained SLAM. *Sensors*.

Liszewski, A. (2023, June 8). *The First Robot Vacuum That Can Climb and Clean Stairs Could Be a Game-Changer*. Retrieved from yahoo! news: https://news.yahoo.com/first-robot-vacuum-climb-clean-181500394.html

Patle, B. a. (2019). A review: On path planning strategies for navigation of mobile robot. *Defence Technology*.

Ramalingam, B., Elara Mohan, R., Balakrishnan, S., Elangovan, K., Félix Gómez, B., Pathmakumar, T., . . . Baskar, C. (2021). sTetro-Deep Learning Powered Staircase Cleaning and Maintenance Reconfigurable Robot. . *Sensors*.

ReportLinker. (2023, March 08). Retrieved from globenewswire: https://www.globenewswire.com/news-release/2023/03/08/2623184/0/en/Cleaning-Robot-Global-Market-Report-2023.html

Sörme, T. E. (2018). *A Comparison of Path Planning.*

T. Kakudou, K. W. (2011). Study on mobile mechanism for a stair cleaning robot - Design of translational locomotion mechanism. *11th International Conference on Control, Automation and Systems* (pp. 1213-1216). IEEE.

Vadakkepat, R. Z. (2003). *Motion Planning of Biped Robot Climbing Stairs.*

Wu, G. L. (2022). Design and Study of a Stair Climbing Robots with Two Wheels and a “4R+2P” Pattern. *Machines*.

Zhang, L. &. (2016). A New Compact Stair-cleaning Robot. *Journal of Mechanisms and Robotics*.