IC 201P – Design Practicum

GROUP 22

Dead Reckoning using Optical Sensors

Aditya Anurag (B21177)
Himanshu Meena (B21408)
Shrisat Ravi Purushottam (B21132)

Kaushik Vishwakarma (B21145)
Atharva Santosh Raut (B21281)
Sher Thaniya (B21166)

Under the supervision of

Dr. Jinesh C. Machchhar, jinesh@iitmandi.ac.in

Dr. Tanushree Parsai, tanushree@iitmandi.ac.in

Indian Institute of Technology Mandi

Certificate

This is to certify that the work contained in the project report entitled "**Dead Reckoning using Optical Sensors(DROS)**", submitted by Group 22 to the Indian Institute of Technology Mandi, for the course IC 201P – Design Practicum, is a record of bonafide research works carried out by him under our direct supervision and guidance.

17/05/2023

Dr. Jinesh C. Machchhar Signature and Date

17.5.23

Dr. Tanushree Parsai Signature and Date

Acknowledgements

We would like to express our sincere gratitude to Pranav Adlinge, who always helps and guides us throughout the project. Special thanks to the PRT lab Laser cutting assistant for cutting the acrylic sheet for us. We would also like to thank the Robotronics club for always being there whenever we needed any general electronics. Finally, we express our gratitude to our mentors for guiding us and always being there, and to IIT Mandi for providing such a wonderful opportunity to bring our imaginations to reality and funding us.

Abstract

Dead reckoning is a method used to determine one's position by utilizing a previously established position and considering the distance and direction traveled since that position was last determined. Dead reckoning systems find extensive application in navigation systems. For instance, they are employed when the GNSS system of a satellite is nonfunctional. Many automated robots also rely on dead reckoning to ascertain their positions, such as room cleaning and grass cutting robots. However, employing high-quality sensors increases the cost of the product, and some products have limitations, such as functioning exclusively on smooth and horizontal surfaces. Therefore, our project aims to develop a cost-effective dead reckoning system capable of operating on even surfaces using two high-DPI optical mouse sensors and a Raspberry Pi. We have selected 10,000 DPI mouse sensors, which are ideal for tracking small movements, while the Raspberry Pi provides a robust and flexible platform for data processing and analysis.

The project will involve the design and construction of a prototype system, which will include two gaming mice, a Raspberry Pi, DC motors, motor drivers, and other necessary components. The system will undergo calibration and testing using a ground truth reference system, such as a motion capturing system, to assess its accuracy and performance. This system holds potential for various applications, particularly in the field of robotics, where precise tracking of a robot's position is essential for effective navigation and obstacle avoidance. For instance, it can be implemented in self-cleaning robots that require knowledge of their position to thoroughly clean every corner of a room, as well as in tasks involving navigation and obstacle avoidance.

The outcomes of the project will not only make significant contributions to the advancement of robotic technology but also push the boundaries of dead reckoning and navigation as a whole.

Contents

ABSTRACT 4

INTRODUCTION 09-11

MARKET RESEARCH 12-14

CONCEPTUAL DESIGN 15-17

EMBODIMENT AND DESIGN 18-30

- Product Architecture
- Detailed Design
- Software Integration
    1. Algorithm
    2. Calibration
    3. Programming Language and packages
    4. Electronic Integration
- Mechanical Aspects

FABRICATION AND ASSEMBLY 31-44

- Bill of materials
- Manufacturing cost per unit
- Mechanical drawing
- Manufacturing Process Description
- Assembly
- Limitations and challenges
- Scheduling plan
- Contribution
- Conclusions
- Readings

REFERENCES 45

List of Figures

Fig 4.1 - Product architecture

Fig 4.2 - Optical Mouse

Fig 4.3 - Raspberry Pi

Fig 4.4 - Motor Driver

Fig 4.5 - Lipo Battery

Fig 4.6 - Power bank

Fig 4.7 - SD card

Fig 4.8 Mouse Movement

Fig 4.9 100 cm in y direction

Fig 4 .10: 90 degree mouse movement graph

Fig 4.11 2 circular motion graph

Fig 4.12 -2 Motor Driver GPIO pins connection

Fig 4.13 - RPI GPIO PIN DIAGRAM

Fig 4.14 Design model 3 D views

Fig 4.15 - Design model 2 D views

Fig 5.1 Acrylic sheet bottom

Fig 5.2 Acrylic sheet mouse mounting plate

Fig 5.3Acrylic sheet Top

List of Tables

Table 3.1 - Decision matrix of Problem Statement

Table 4.1 - DPI reading while Right Mouse calibration

Table 4.2 - DPI reading while Left Mouse calibration

Table 5.1 - Billing of materials

Table 5.2 - Manufacturing cost details

Table 5.3 - Scheduling Plan

Table 5.4 - Reading for moving 100 cm in y-axis only

Table 5.5 - Reading for moving 90 degrees

Table 5.6 - Reading for two circular motion on floor

Abbreviations

DR - Dead Reckoning

DRS - Dead Reckoning System

DROS - Dead Reckoning using Optical Sensors

RPI -Raspberry PI

DPI - Dots per inch

wrt - with respect to

ΔxR - movement of right mouse in x-axis

ΔyR - movement of right mouse in y-axis

ΔxL - movement of left mouse in x-axis

ΔyL - movement of left mouse in y-axis

Chapter 1

Introduction

Dead reckoning, also known as deduced reckoning or DR, is a method used in navigation to estimate the current position of a moving object based on its previously known position and the course and speed at which it is traveling. The term "dead" in dead reckoning comes from the fact that it does not take into account external factors such as wind, currents, or other influences that may affect the object's actual position.

Dead reckoning relies on the principle of using a known starting point, along with measurements of the object's speed and direction, to calculate its subsequent positions. By continuously updating the position based on these measurements, an approximate track can be determined.

Dead reckoning is commonly used in various forms of transportation, including aviation, maritime navigation, and land-based navigation. It serves as a basic means of navigation when other methods, such as celestial navigation or GPS, are unavailable or unreliable. In modern times, it is often supplemented or replaced by more accurate positioning systems like GPS, which provide real-time and highly precise location information.

1. **Background of the problem:**

   The existing solutions for dead reckoning primarily rely on the kinematics of the driving wheels employed in the devices. However, these solutions encounter certain challenges. For instance, the encoders utilized in these systems struggle to accurately estimate the distance traveled in the presence of slip or external factors causing push, as well as potential miscalculations arising from uneven surfaces, among other issues.

2. Scope of the problem

The solution we are providing is based on a pair of optical mice rigidly connected to the robot body. The main advantages of this approach are:

a. The measurements provided by the optical mice are not susceptible to slipping, as they operate independently of the traction wheels. Furthermore, they are not affected by crawling, as they have the ability to measure displacements in any direction. This makes the optical mice a reliable and versatile solution for dead reckoning, overcoming the limitations associated with slipping and crawling in traditional methods.
b. This localization system operates independently of the robot's kinematics.
c. Our cost-effective solution utilizes data from optical mice, which provide real-time measurements of changes in both the x and y directions. This enables us to calculate the angle by which the system is rotated with respect to its original orientation.

3. Design philosophy used in this report

The design philosophy of this project is centered around developing a practical and reliable dead reckoning system (DRS) by leveraging optical sensors and Raspberry Pi (RPI) for precise and dependable position tracking. The approach follows an iterative process, focusing on optimizing and refining the system to enhance its performance while ensuring functionality and reliability. The system's design aims for versatility, making it suitable for application across different industries and use cases.

4. Problem statement

The goal of this project is to develop a dead reckoning system that utilizes optical sensors to achieve accurate and reliable real-time tracking of a robot's movement and position.

5. Beneficiaries (Intended market)

The beneficiaries of this project span a wide range of individuals and organizations across various industries that demand accurate and reliable position tracking. The primary market for the project is expected to be in the robotics industry, where precise positioning is essential for tasks such as navigation, movement control of artificial robots, obstacle avoidance, and other applications. Some potential beneficiaries of the project include:

1. **Robotics companies**: Companies that manufacture and develop robots can benefit from the project by implementing the DRS developed in this project to improve the accuracy and reliability of their robot's position tracking.
2. **Automation industry:** To implement the system in autonomous bots that require accurate positioning for navigation.
3. **Mapping** : The system can be used to track the position of obstacles in a fixed area to draw its map if autonomous movement is provided for the swiping area.
4. **Miscellaneous**: Pathfinding (finding a path from the current location to a target location), coverage (covering the entire floor area with minimal repetition), and creating easy-to-use and entertaining indoor robots for children are potential applications within the toy industry.

Chapter 2

Market Research

1. **Existing products in the market.**

Most localization robots primarily rely on dead reckoning based on odometry to perform their navigation tasks. This method is often used independently or in conjunction with other absolute localization systems (Borenstein and Feng, 1996). Odometry typically involves measuring the space covered by the robot's wheels using encoders. These encoders can be positioned directly on the wheels or on the engine axis. The measurements obtained from the encoders are then combined to calculate the robot's movement along the x and y-axes, as well as its change in orientation.

It is well-known that odometry is subject to several limitations and challenges, including:

- systematic errors caused by factors such as unequal wheel diameters, imprecisely measured wheel diameters and wheel distance, or imprecisely measured tread (Borenstein and Feng, 1996);
- Non-systematic errors caused by irregularities on the floor, bumps, cracks, or wheel slippage.

2. **Present a comparison with the existing products/similar technologies that exist in the market.**

The Telenav Scout GPS Link utilizes odometry, rather than pure dead reckoning, to provide location information in areas with weak or unavailable GPS signals. Odometry, in this case, is used to estimate the vehicle's position by measuring the rotation of the wheels and the distance traveled.

The HG4930 IMU (Inertial Measurement Unit) is indeed a highly accurate sensor system that employs gyroscopes and accelerometers to measure the angular rate and acceleration of a vehicle. With these measurements, the IMU can analyze the vehicle's motion and calculate its orientation and movement over time. By utilizing this information, the IMU can provide estimations of the vehicle's position.

3. Identify what are the problems associated with the existing alternatives.

One significant challenge in dead reckoning navigation is the accumulation of errors over time. This is particularly true in situations where the vehicle operates in areas with weak GPS signals or variable driving conditions. As the errors in position estimation accumulate, there is a higher likelihood of inaccuracies and potential navigational errors. It is crucial to account for and mitigate these accumulated errors through calibration, error correction algorithms, and periodic recalibration using absolute localization systems to maintain accurate positioning.

Limited accuracy:

Indeed, despite utilizing odometry and other dead reckoning techniques, the accuracy of position estimates is often limited, particularly in areas with complex terrain or significant interference from other signals. Factors such as uneven surfaces, variations in wheel traction, and external disturbances can introduce errors and reduce the reliability of dead reckoning-based positioning. In such challenging environments, it becomes crucial to integrate additional localization methods, such as GPS, visual odometry, or sensor fusion approaches, to improve the overall accuracy and robustness of the position estimates. These complementary techniques help compensate for the limitations of dead reckoning and enhance the navigation capabilities in complex and interference-prone settings.

Reliance on external signals:Many dead reckoning navigation systems rely on external signals, such as GPS or wheel rotation sensors, which are susceptible to interference or disruption. This vulnerability can result in inaccuracies or the loss of position information.

High cost: Some dead reckoning navigation systems, particularly high-end inertial measurement units, can be costly and may not be feasible for all applications due to their price.

4. Mention how your intended product stands different from the existing product.

We propose a novel dead reckoning method that demonstrates exceptional robustness against non-systematic errors by utilizing odometric sensors that are decoupled from the driving wheels. Our approach involves incorporating two optical mice, securely fixed to the underside of the robot, to gather measurements. By employing two mice, we obtain four measurements, allowing us to estimate three parameters (Δx, Δy, $\Delta \theta$). Although only three of these measurements are independent due to the fixed position of the mice, the fourth measurement can be derived from them. We have specifically chosen optical mice over traditional ones because they can be used without direct contact with the floor. Consequently, we eliminate issues such as constantly pressing the mouse against the ground, problems arising from friction, and complications associated with dust accumulation in mouse mechanisms.

Chapter 3 - Conceptual Design

Our experience in the design practicum course was transformative as it provided us with the chance to connect with our classmates and gain valuable knowledge. Through engaging in lively discussion, we explored numerous ideas, some of which we reached agreements on, while others were dismissed. We also encountered conflicting viewpoints during this process. To determine the most promising idea to pursue, we held multiple brainstorming sessions and continued to do so even after idea selection, using it to troubleshoot challenges and make decisions.

In this chapter, we reflect on our journey in the design practicum, focusing on our experiences in idea generation, proposition, and selection. These aspects have played a pivotal role in enriching our overall learning experience.

1. **Ideation**
   - We initiated the ideation process by identifying 100 prevalent problems in our surroundings or those commonly encountered by people.
   - Each team member actively contributed their top 5 problems, which were subsequently compiled and documented for reference and further exploration during the ideation process.
   - Next, we embarked on the elimination process, filtering out problems that were beyond our scope or required significant funding. Additionally, we excluded problems that lacked potential for significant social impact.
   - After preparing a comprehensive presentation, we engaged in a discussion with our mentors, Dr. Jinesh C. Machchhar and Dr. Tanushree Parsai. They meticulously analyzed all aspects of the identified problems and offered guidance and their own ideas for a potential project concept.
   - Once we had identified all the problems, we ultimately made the decision to focus on the problem proposed by one of our mentors, Dr. Jinesh C. Machchhar, which was DROS (Dead Reckoning using Optical Sensors).

2. **Brainstorming and Idea generation**:

- Once we had finalized our problem, we proceeded to formulate a concise problem statement that highlighted the key points and challenges to be addressed. With the guidance of our mentor, we identified potential solutions to tackle the problem.
- We conducted an extensive search for existing solutions by reviewing various research papers provided by our mentor and utilizing the Google Scholar website. This allowed us to gather valuable insights and knowledge on the current state of the field.
- After analyzing the problem statement, we engaged in discussions regarding component availability, potential applications, future scope, cost-effectiveness, and usability. Through careful consideration and evaluation, we arrived at our final solution for the proposed problem statement.

3. Matrix for Problem Statement

	DECISION MATRIX						
	Smart traffic for ambulane	Plant watering system with humudity sensor	Disease Predictor	Tracker Band	Heart rate detector	A machine to fix broken pipes	Dead reckoning using optical sensors
Existing solution	4	3	1	4	3	2	3
Beneficiaries	5	4	4	3	4	2	5
Ease	2	3	3	1	2	2	5
Pros and cons	3	3	4	5	3	3	4
Motivation	4	3	4	3	2	2	4
Cost	5	4	3	4	3	4	3
Time	5	4	3	5	4	5	4
Innovation	5	4	3	2	3	3	5
Final	4	3	3	3	3	2	5
Scale	Rating	Description					
	1	Very less					
	2	Less					
	3	Moderate					
	4	High					
	5	Very high					

Table 3.1 - Decision matrix of Problem Statement

Final Proposed solution:

After multiple meetings, in-depth discussions, and productive brainstorming sessions, we successfully reached a final solution for the problem statement. This solution is the result of our collective efforts and rigorous evaluation of various ideas and possibilities.

In order to develop an accurate localization system for robots, we have implemented a dead reckoning system that relies on measuring the incremental movements of the robot directly from the floor using optical mouse sensors. These sensors allow us to measure the movement along two axes. To accurately calculate the robot's position and orientation deviation, it is essential to attach two optical mouse sensors to the robot. By analyzing the sensor values obtained from these sensors, we have conducted several experiments with the prototype to validate the effectiveness of the algorithm used in the system.

Chapter 4

Embodiment and Detailed Design

1. Product architecture -

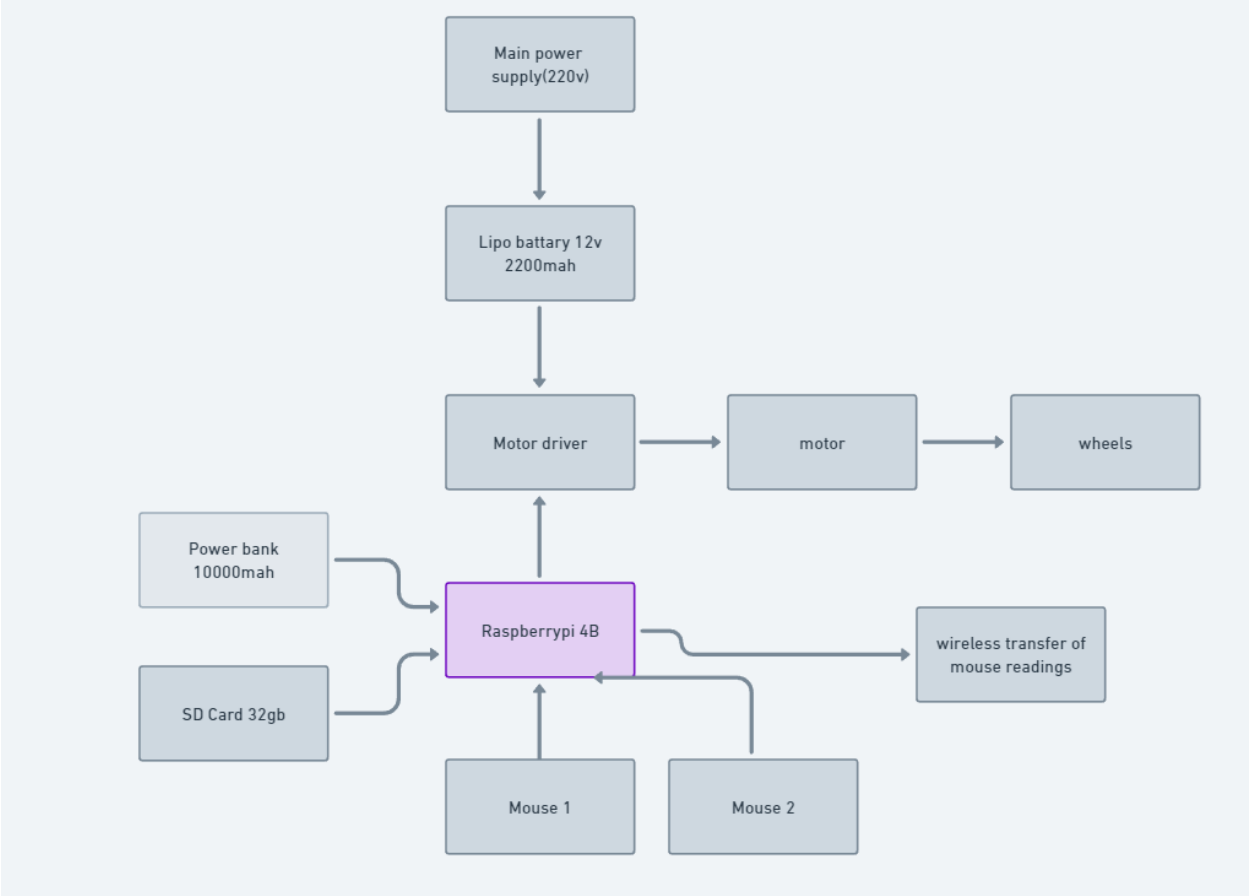


Fig 4.1 Product architecture

2. Detailed design

Electrical/Electronics aspect: These are the specification on which we concluded to use:

- **Optical Mouse**: By utilizing the optical flow sensors found in USB mice, we are able to measure the distance traversed by the robot. However, due to the limitation of distance measurement with a single mouse, we have incorporated two mice into the system. This allows us to calculate not only the distance but also the angle of movement using basic geometric principles. To ensure accuracy in our measurements, we have utilized a high-resolution mouse with a **DPI (Dots Per Inches) value of 10,000**. This enables precise tracking of the robot's movements and enhances the overall accuracy of the localization system.

Fig 4.2 - optical mouse

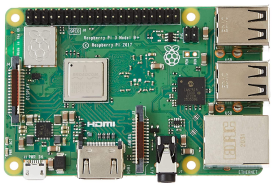


Fig 4.3 - Raspberry Pi

- **Raspberry Pi 3B+:** In order to accurately calculate new coordinates and update them for every inch of movement, we require a powerful processing capability on a single circuit board. This necessitates a system with high computational capacity. Additionally, to handle the calculations efficiently, a larger RAM capacity is preferred. However, due to market shortages and high prices, we are constrained to using a **1GB RAM** module, which may have limitations in handling complex calculations and storing large amounts of data. Despite this limitation, we have optimized our algorithms and code to maximize the performance within the available resources.

- **DC Motor:** In order to strike a balance between speed and accuracy, we carefully considered the capabilities of our system components. We wanted to ensure that the speed of the Bot was not too fast, as it could potentially exceed the capabilities of the optical mouse sensors in providing accurate movement distance measurements. Furthermore, we needed to ensure that our circuit board could effectively handle the computational demands of the system. Taking these factors into account, we made the decision to use **12V DC motors**, which offered a suitable compromise between speed and accuracy for our specific requirements.

- **DC Motor Driver:** To ensure precise control over the motor rotation, we recognized the importance of using a suitable motor driver. The motor driver would enable us to regulate the motor's movement in accordance with the algorithm of the Bot, ensuring that it does not skip any areas or lag behind. After careful consideration, we opted for a motor driver that supported a voltage range of **5-30V.** This range provided us with the flexibility to accommodate small variations in the power supply, allowing for more stable and consistent motor control.

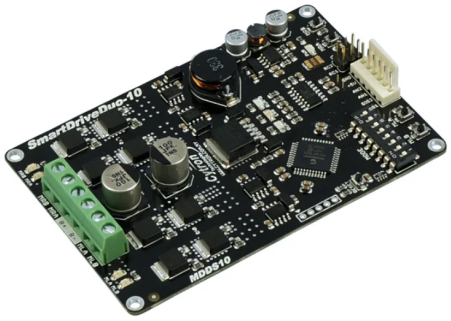


Fig 4.4 - Motor Driver

- **LIPO Battery:** For supplying power to the motor we needed a DC battery of the same rating. Thus, **12V LIPO Battery**.
- **Power Bank:** For providing constant power to Raspberry Pi which is very sensitive to power supply we needed a stable 5V power supply thus we required a **Power Bank**.

Fig 4.5 - Lipo Battery

Fig 4.6 - Power bank

- **SD Card:** The small size of the RAM in the Raspberry Pi posed challenges in terms of software installation and the limitation of not being able to permanently store data. To address this, we opted for a storage memory card with a capacity of 32GB. This SD card provided ample space to store the written algorithm and other necessary data, allowing for smooth operation and efficient execution of the system.

Fig 4.7 - SD Card

- **Acrylic Sheet:** We opted to use acrylic sheets for the chassis design due to their favorable characteristics such as being lightweight, strong, and easy to cut. To meet our specific requirements, we utilized two different thicknesses of acrylic sheets. For the chassis, we used a 5mm thick sheet to provide sufficient strength and durability. Additionally, for mounting the mouse sensors, we chose a 2mm thick sheet to ensure that the sensors would remain in close proximity to the ground and accurately sense the movement without any interference..
- **Miscellaneous:** Now these are some other components which are also part of Assembling ex: Screws, Nuts, Feviquick, Double-Sided tape, Jumper wires, etc.

3. Software Integration

1. Algorithm for Dead Reckoning

- At every iteration of updates 4 instantaneous displacements are received - (ΔxL, ΔyL) and (ΔxR, ΔyR).
- The Algorithm should be able to calculate the X, Y and θ with respect to the origin of the global coordinate system.

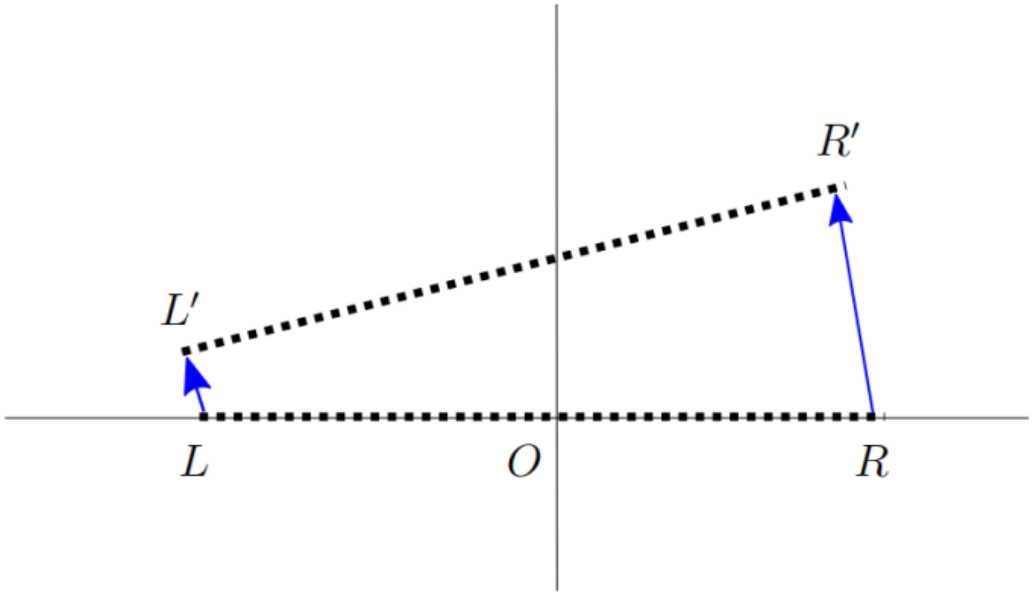


Fig 4.8 Mouse Movement

In Fig 4.8, the initial position of the mice is indicated by the line LR.

The point L is the location of the left mouse, similarly for point R. After one iteration of update the new position of the mice is indicated by the line L'R'. All the calculations are done with respect to the center of the line joining two mice. Let Xu, Yu, θu be the updates in the local coordinate system then they are calculated as -

$$Xu = (\Delta XR + \Delta XL)/2 \qquad (2.1)$$

$$Yu = (\Delta YR + \Delta YL)/2 \qquad (2.2)$$

$$\Theta u = \text{atan2}(\Delta YR - \Delta YL, \Delta XR - \Delta XL + D) \qquad (2.3)$$

where D is the distance between the two mice sensors and atan2 calculates the inverse tan operation and outputs angle from $-\pi$ to π.

Proof for equations 2.1 and 2.2

Coordinates of L and R are (-D/2, 0) and (D/2,0) respectively. With a change of (ΔXL, ΔYL) in left mouse's position and (ΔXR, ΔYR) in right mouse's position, the coordinates of L' and R' are (-D/2+ΔXL, ΔYL) and (D/2+ ΔXR, ΔYR). To find the position of the center of L'R', we take the average of co-ordinated of L' and R'.

Proof for Equation 2.3

The change in the orientation of the line joining the two mice is tan inverse of the change in y-direction by change in x-direction

$$R'L' = R' - L'$$

$$\therefore R'L' = (\Delta XR - \Delta XL + D, \Delta YR, \Delta YL)$$

$$\theta u = \text{atan2}(R'L'_y , R'L'_x)$$

Conversion of Local coordinate system updates to Global coordinate system update-

Let $(Xi+1, Yi+1, \theta i+1)$ be the coordinates in GCS at $(i + 1)$th iteration and $(Xi , Yi , \theta i)$ be the coordinates in GCS at (i) th iteration then -

$$\boldsymbol{\theta_{(i+1)} = \theta_i + \theta u}$$

$$\begin{pmatrix} X_{i+1} \\ Y_{i+1} \end{pmatrix} = \begin{pmatrix} X_i \\ Y_i \end{pmatrix} + \begin{pmatrix} cos\theta_i & sin\theta_i \\ -sin\theta_i & cos\theta_i \end{pmatrix} \begin{pmatrix} X_u \\ Y_u \end{pmatrix}$$

The θu needs to be simply added to the current orientation of the robot with respect to the global coordinate system. The local coordinate system is currently at an angle of θi with the global coordinate system, hence to get the X and Y updates in GCS we need to rotate the updated (Xu,Yu) by negative of θi .

That rotation matrix is multiplied by the vector (Xu,Yu) which is added to the current (Xi ,Yi) to obtain the updated position.

☐ **100cm in y direction.**

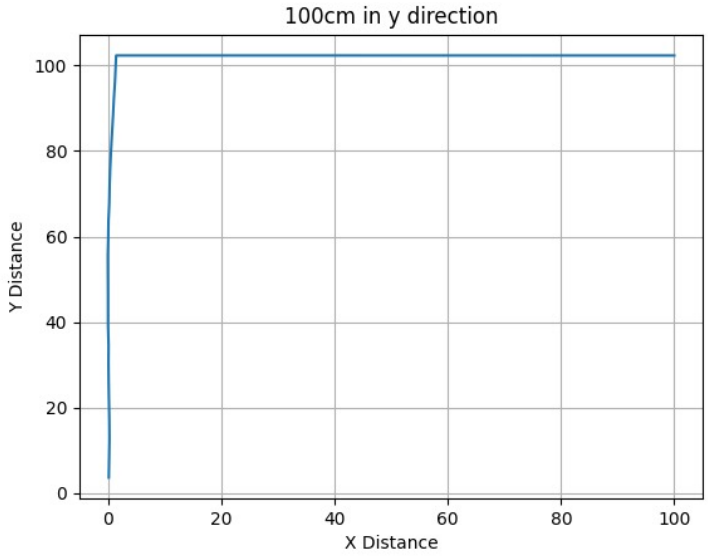


Fig 4.9 100 cm in y direction

☐ **90 degree movement**

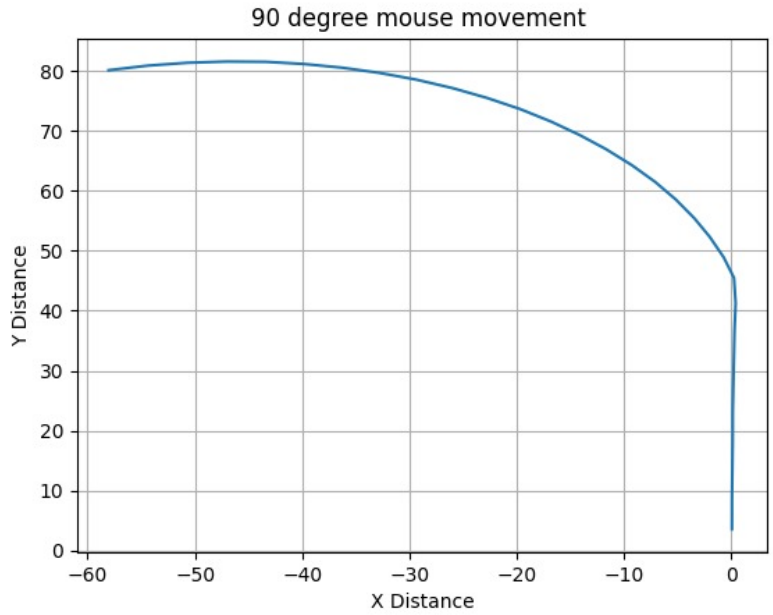


Fig 4 .10: 90 degree mouse movement graph

☐ **2 circular motions on floor**

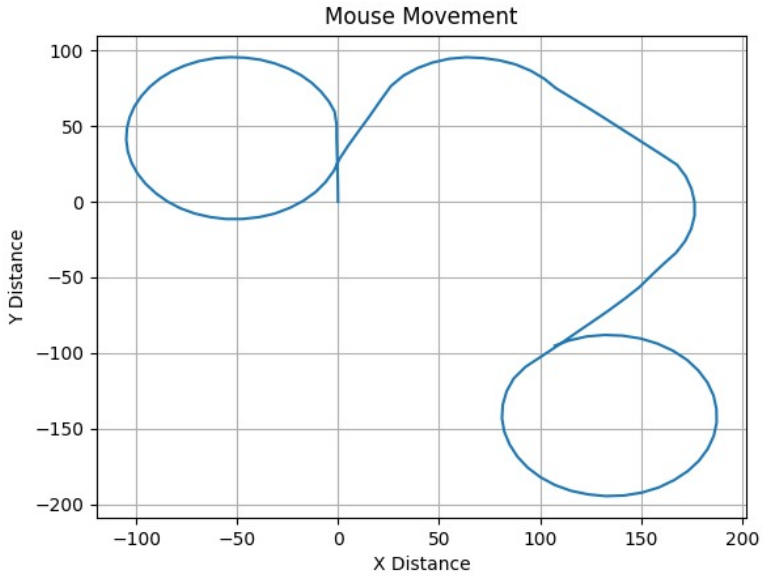


Fig 4.11 2 circular motion graph

All above Graph made by using data saved by robot in csv by moving bot with keyboard control

2. Calculation of Mouse DPI(Dots per linear Inch)

Steps to calculate DPI:

- Make a 20 cm horizontal line on graph paper.
- Connect mouse with laptop and Run this code Link
- Move the mouse along the horizontal line and note the Dp centimeter of the mouse for 20 cm.
- Repeat above steps and note down Dp Centimeter for 5 times with both mouses.
- Take the average of both mouse Dp centimeters separately and Divide by 20 to get Dp centimeters for 1 cm.
- As 1 Inch = 2.54 cm ,multiply Dp centimeter with 2.54 to get DPI of the mouse.

Calculation :

For Right mouse :

No	**Readings**
1	79589
2	81040
3	82207
4	79426
5	79390
sum	401652/5=80330.4

Table 4.1 - DPI reading while Right Mouse calibration

1 inch = 2.54 cm
80330.4/20 = 4016.5 Dp Centimeter
4016.5 * 2.54 = 10201.5

No	Readings
1	82056
2	81866
3	81946
4	81080
5	80346
sum	407294/5=81458.8

Table 4.2 - DPI reading while Left Mouse calibration

81458.8/20 = 4072.94 Dp Centimeter
4072.94 * 2.54 = 10345.2676

3. Programming language and Packages Used

Complete code can be found on github Link

Programming language : We have used Python 3.10.11

Python Packages :

Evedev : Evedev is python package available for Linux based operating Systems. We have used this package to get mouse change in X and change in Y.

Math and Numpy : We have used these packages for calculation .

Steps for running code;

- Run **evtest** in terminal to know the event files with which mouse connected and change it in code for both mouse.
- Run code file by writing **python3 filename** in terminal.

Use keyboard and Sockets: These packages are used for building wireless connection with raspberry pi and laptop. In this way we can send singles of "W"

,“A”, “S”, “D” to raspberry pi using a laptop. So that we can control our bot using laptop inputs.

4. RPI Integration

- Flash the raspberry with an sd card and install raspbian os in it.
- Installed **puTTy** software for connection with RPI wifi

  First the RPI connects with the mobile phone’s hotspot and then mobile connects with the laptop.

- Used **VNC viewer** to run program and make necessary changes

  On a RPI using a laptop. No HDMI cable or ethernet cable are used.

- RPI takes all the input from the mouse and does all the calculations.

5. GPIO pins and electric connects.(electronics integration)

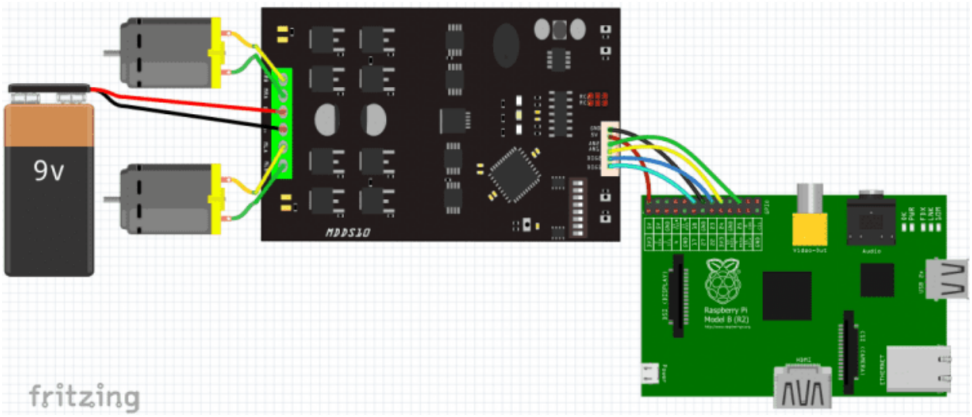


Fig 4.12 -2 Motor Driver GPIO pins connection

- We used GPIO (PIN NO. 6) for ground.
- GPIO (PIN NO. 12) for DIR1.
- PIN NO.16 for DIR2.
- PIN NO.18 for PWM1.
- PIN NO. 22 for PWM2.

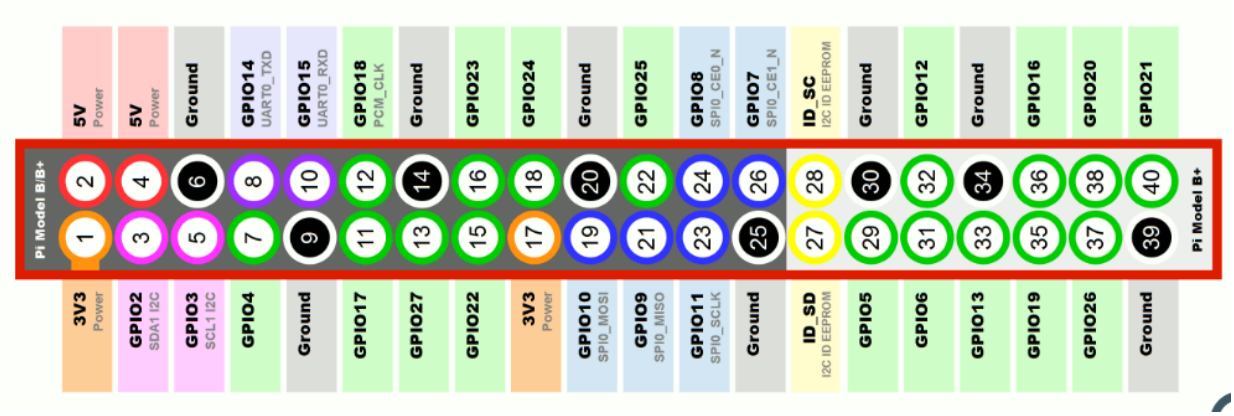
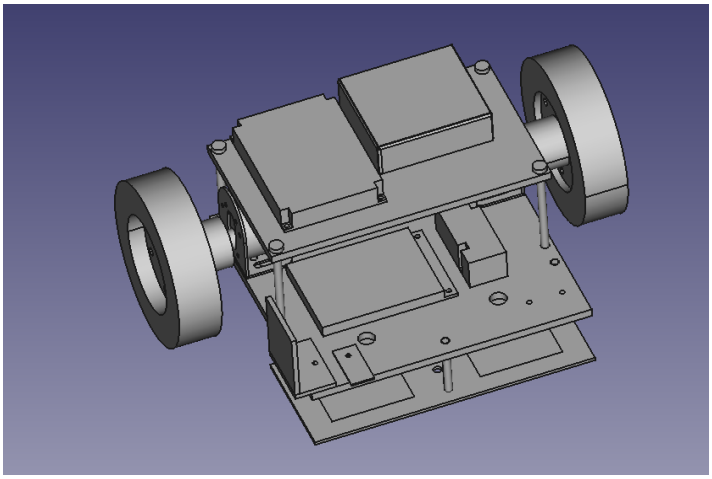


Fig 4.13 - RPI GPIO PIN DIAGRAM

4. Mechanical Aspects 3-D view-



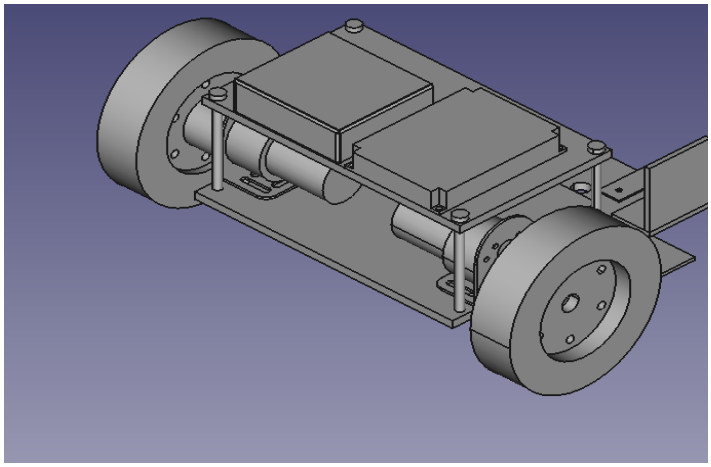
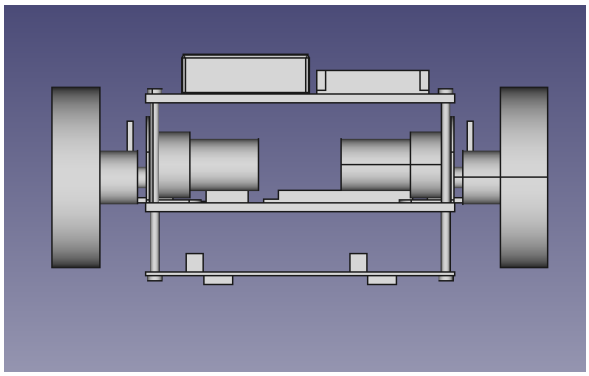
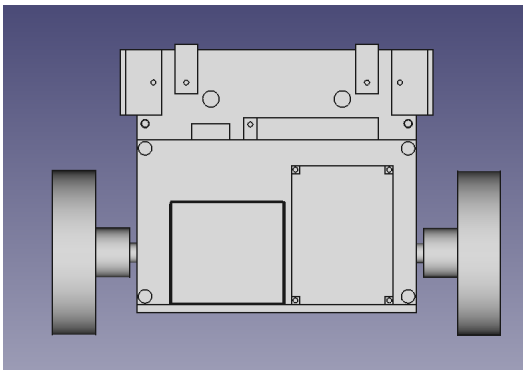


Fig 4.14 Design model 3 D views

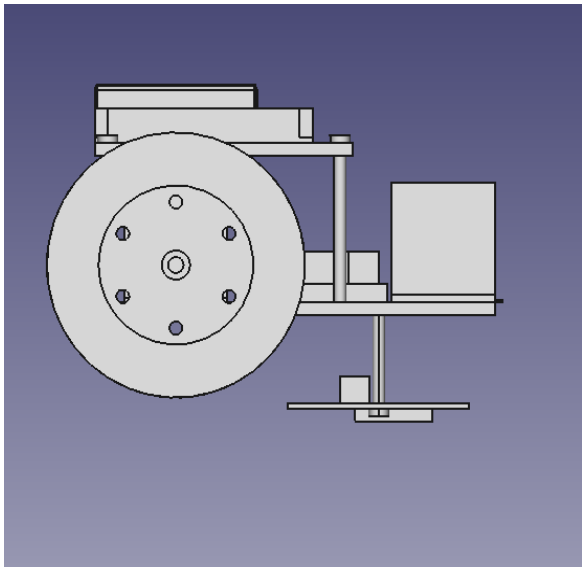
Orthographic view



1.Front view



2. Top view



3. Side view

Fig 4.15 - Design model 2 D views

Chapter 5
Fabrication and Assembly

1. Bill of Materials (BOM):

S.No.	Item	Quantity	Price(in Rs.)
1	RPI	1	10074
2	Gaming Mouse(for optical sensors)	2	2698
3	Cytron MDD10A Dual Channel Enhanced 10Amp DC Motor Driver 30A Peak	2	3936
4	EasyMech Anti slip Motor Coupling	2	1158
5	BreadBoard	1	61
6	M to F, F to F, M to M	1	133
7	Ultrasonic Sensor	2	122
8	Disc wheel	2	2090
9	DC Motor 12 V	2	952
10	Motor Brackets	2	598
11	Invento 2200mah LIPO battery	2	2850+1399
12	MI power bank	1	1199
13	Screw Driver Set	1	149
14	Sandisk 32GB SD Card	1	369
15	Acrylic sheet(2mm)	1	264
16	Acrylic sheet(5mm)	1	519
17	Nut and Bolt	70	180
18	Glue Gun	1	352
19	Report Print	1	180
20	Fevi quick	15	225
21	Lipo charger	1	500

Table 5.1 Bill of materials

Total : 30000 RS Due to shortage of RPI we have to buy 2000 RS RPi in 10000 RS

2. Manufacturing Cost per unit: Rs. 12,000

S.No.	Item	Quantity	Price(in Rs.)
1	RPI(cost without shortage)	1	2000
2	Gaming Mouse(for optical sensors)	2	2698
3	Cytron MDD10A Dual Channel Enhanced 10Amp DC Motor Driver 30A Peak	1	1968
4	EasyMech Anti slip Motor Coupling	2	1158
5	F to F	1	133
6	Disc wheel	2	2090
7	DC Motor 12 V	2	952
8	Motor Brackets	2	598
9	Invento 2200mah LIPO battery	1	2850
10	MI power bank	1	1199
11	Screw Driver Set	1	149
12	Sandisk 32GB SD Card	1	369
15	Acrylic sheet(2mm)	1	264
16	Acrylic sheet(5mm)	1	519
17	Nut and Bolt	70	180
18	Total		17127
19	If components taken at wholesale price(manufacturing cost)		12000

Table 5.2 - Manufacturing cost details

3. Mechanical drawing:

1. Chassis:length (170x160mm)

   thickness(5mm)

a. Material: Acrylic
b. Manufacturing Process: laser cutting.
c. Drawing:

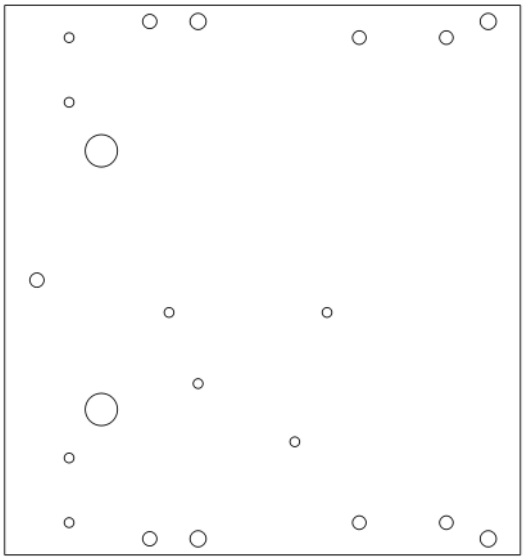


Fig 5.1 Acrylic sheet bottom

2. Bottom:length (170x70mm)

   thickness(2mm)

- Material: Acrylic sheet
- Manufacturing Process: laser cutting

o Drawing:

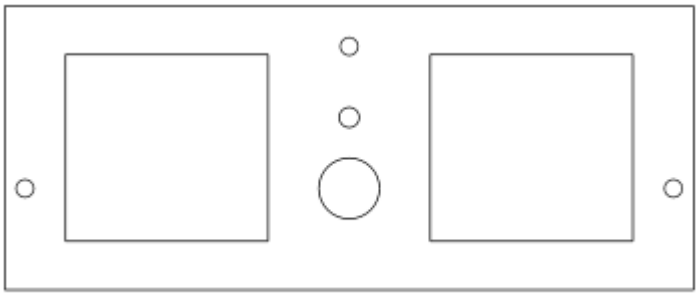


Fig 5.2 Acrylic sheet mouse mounting plate

3.Floor:length (170x100mm)

thickness(5mm)

o Material: Acrylic sheet

o Manufacturing Process: laser cutting

o Drawing:

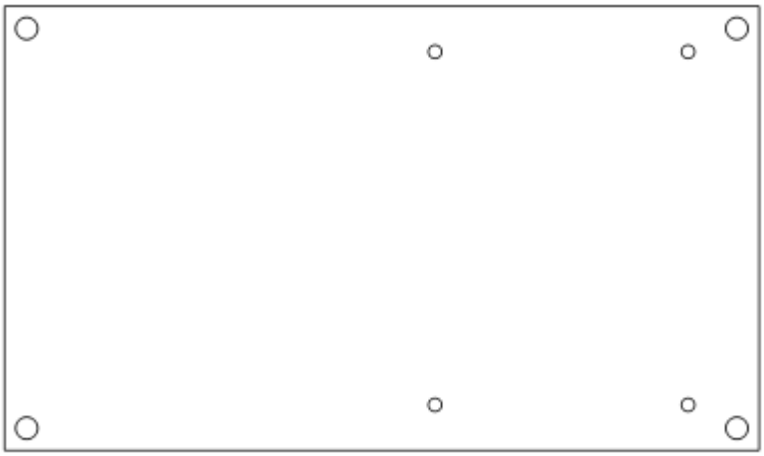


Fig 5.3Acrylic sheet Top

4. Manufacturing Process description:

For the fabrication part only we needed Acrylic sheet cutting rest of the items were to be assembled properly in their place.

5. Assembly

- We attached chassis with L-Brackets on which our DC motor was to be Mounted.
- Then fastened the DC motor with the help of screws and Nut-Bolts on L brackets.
- After which we fastened wheel anti slip couplings with wheels.
- Then those coupling were fastened on DC motors.
- Doing so our Lower Main chassis was ready now this main chassis was attached with upper(on which Raspberry and Motor driver were fixed with Double sided tape) and for lower we directly attached mouse with main chassis using feviquik and glue gun.
- For fixing lower chassis there were some cautions:
    - ❖ Distance of the chassis should not be too high for optical flow sensors to detect ground.
    - ❖ Calibration of distance between two optical flow sensors(on which angular calculation of the bot movement depends).

6. Limitations and Challenges:

a. Limited accuracy: Sometimes the optical sensor may not be able to provide the accurate positions

b. Vulnerability to external factors: DRS can be affected by external factors such as vibrations, changes in surface texture, and obstacles leading to errors in positioning

c. Inaccuracy in angle calculation: Due to improper calibration of DPI of the mice, it was difficult to calculate the accurate angle, i.e. rotation of the prototype

d. Movement of the bot: Faced difficulty in estimating and feeding the appropriate values to move the bot to desired position.

7. Scheduling plan:

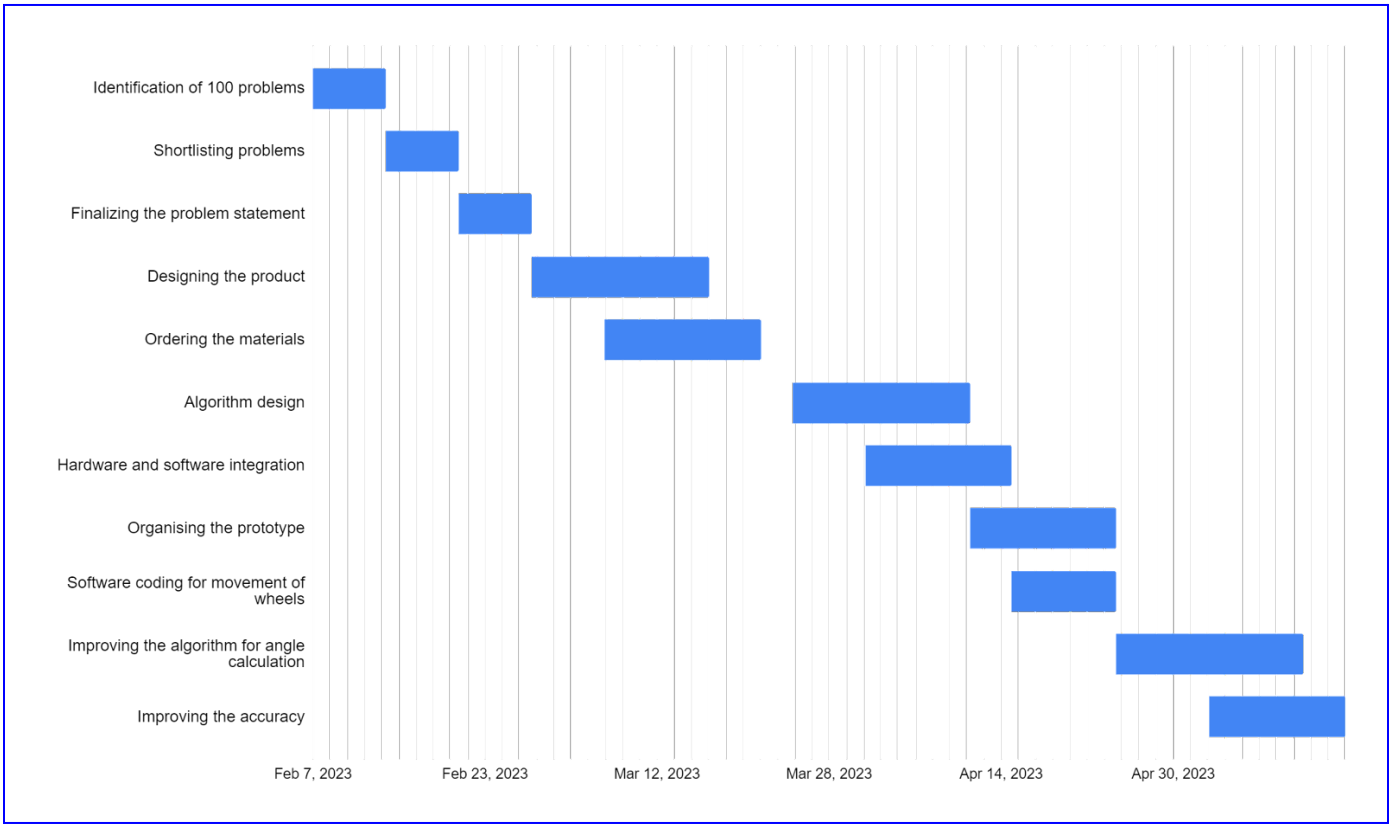


Table 5.3 - Scheduling Plan

8. **Contribution:** The contribution of each team member:
   - Aditya Anurag : Assembly of the bot, Circuit and electronics part and algorithm for improving accuracy by using different DPI for each mouse.
   - Kaushik Vishwakarma : Coding in Raspberry Pi and algorithm part . implement movement of wheel with laptop keyboards using socket python package . Testing and identifying limitations of robots .Making 2 D graph with points taken by robot while moving . Documented Report .
   - Shrisat Ravi Purushottam : Coding in Raspberry Pi and algorithm part . implement movement of wheel with laptop keyboards using socket python package . Testing and identifying limitations of robots .Making 2 D graph with points taken by robot while moving .

- Sher Thaniya : Searching and buying components with specifications, Documenting Report, helped in mouse mounting and testing the prototype.
- Himanshu Meena : CAD modeling and fabrication
- Atharva Santosh Raut : CAD modeling, fabrication .

9. Conclusions:

Using optical sensors we have derived a method which can accurately give configuration estimates of a robot positioning. The optical sensors provide the relative movement between two range sensor readings. Using the appropriate algorithm robot configuration is successfully estimated.

10. Readings of movement of the bot for desired motion

☐ **Readings we get for moving in a straight line for 100cm in y direction.**

count	x distance	y distance	theta(θ)
1	0.115987	3.609493	0.412522
2	0.179118	8.149266	0.169919
3	0.238984	12.73789	0.119016
4	0.21085	17.34834	-0.32
5	0.141392	21.86061	-0.64789
6	0.088149	26.23593	-0.91661
7	0.057628	30.47435	-0.9198
8	0.076939	34.66149	-0.77251
9	-0.01639	38.86811	-0.76138
10	-0.02339	43.04928	-0.40983
11	-0.02768	47.2958	-0.01481
12	-0.04745	51.4542	0.209513
13	-0.05612	55.60918	0.339183

14	0.012221	59.56356	0.732177
15	0.0573	63.68202	0.879914
16	0.201045	67.77219	1.412449
17	0.266517	71.88611	1.52158
18	0.363728	75.97291	1.557269
19	0.506131	80.00471	1.797057
20	0.683408	84.04847	2.149917
21	0.867982	88.21096	2.244474
22	1.014718	92.23428	2.390801
23	1.209735	96.2133	2.457604
24	1.345547	100.2813	2.202073
25	1.410035	102.3486	2.208238
26	1.410035	102.3486	2.208238
27	1.410035	102.3486	2.208238
28	1.410035	102.3486	2.208238
29	100	102.34	2.2

Table 5.4 - Reading for moving 100 cm in y-axis only

☐ **Reading we get for moving the mouse 90 degrees.**

count	**x distance**	**y distance**	**theta**
1	0.040941	3.576252	0.203656
2	0.045186	8.615382	0.42532
3	0.079108	13.60038	0.59764
4	0.109608	18.49983	0.585606
5	0.116042	23.18871	0.532548
6	0.166067	27.71114	0.743381

7	0.223724	32.23305	0.917798
8	0.279907	36.763	0.780884
9	0.384398	41.26447	0.980166
10	0.231132	45.48632	0.162578
11	-0.73274	48.92398	-3.69482
12	-2.02274	52.30958	-7.87948
13	-3.45618	55.4116	-11.7732
14	-5.19333	58.55822	-16.0917
15	-7.12638	61.50263	-20.2121
16	-9.2731	64.26927	-24.6041
17	-11.6895	66.94719	-29.0565
18	-14.1474	69.3088	-33.1108
19	-16.808	71.54901	-37.0864
20	-19.6563	73.613	-41.0537
21	-22.7642	75.48784	-45.4674
22	-25.9488	77.11617	-49.6685
23	-29.3609	78.5461	-54.0957
24	-32.8448	79.68819	-58.6286
25	-36.3124	80.55029	-63.0803
26	-39.8792	81.16387	-67.4352
27	-43.3882	81.51903	-71.493
28	-47.0221	81.56896	-75.7864
29	-50.705	81.36263	-80.1827
30	-54.3821	80.89075	-84.5918
31	-58.046	80.12611	-89.0925

Table 5.5 - Reading for moving 90 degrees

☐ Readings of 2 circular motions on floor

count	x distance	y distance	theta
1	0	0	0
2	-0.03093	4.237273	-0.09872
3	-0.12843	13.81083	0.003063
4	-0.29412	23.24358	-0.19572
5	-0.48045	32.55814	-0.23312
6	-0.69386	41.66763	-0.13896
7	-0.72411	51.13076	0.081506
8	-1.69281	59.59645	-3.6408
9	-4.57856	66.38719	-11.8075
10	-8.46826	72.85046	-20.3297
11	-13.0429	78.62652	-28.5029
12	-18.4678	83.76746	-36.565
13	-24.5936	88.01805	-44.32
14	-31.3389	91.41382	-52.379
15	-38.3176	93.8404	-60.3407
16	-45.8592	95.20828	-68.9223
17	-53.372	95.54214	-77.3404
18	-60.6445	94.95311	-85.1629
19	-68.7468	92.984	-93.8874
20	-75.86	90.0074	-102.044
21	-82.2817	86.24978	-110.196
22	-87.9383	81.67484	-118.651
23	-93.1237	76.0718	-126.972

24	-97.3318	69.89853	-135.05
25	-100.799	63.05703	-143.312
26	-103.178	55.81412	-151.788
27	-104.518	48.40299	-159.605
28	-104.858	40.85635	-167.756
29	-104.076	33.17266	-175.678
30	-102.218	25.71193	-183.557
31	-99.3266	18.69781	-191.688
32	-95.349	11.93314	-199.838
33	-90.2218	5.607641	-209.115
34	-84.5479	0.231984	-216.999
35	-78.1132	-4.30483	-225.315
36	-71.0681	-7.73961	-233.562
37	-63.3816	-10.1261	-241.99
38	-55.4438	-11.4121	-250.7
39	-47.2003	-11.3903	-259.856
40	-38.9058	-10.1524	-268.742
41	-31.2761	-7.80917	-277.149
42	-23.8164	-4.04581	-285.843
43	-17.0928	0.693791	-294.699
44	-11.1558	6.377031	-303.724
45	-6.18245	12.96682	-312.548
46	-2.12785	20.34959	-321.795
47	0.732036	28.31259	-330.605
48	5.165716	37.33053	-332.039
49	10.2337	46.72032	-331.385
50	15.6856	56.70515	-331.683

51	20.95196	66.78751	-331.286
52	26.2115	76.49645	-331.028
53	32.45162	83.39764	-323.019
54	39.34287	88.33345	-314.403
55	46.92467	92.04394	-305.015
56	55.09937	94.50185	-295.28
57	63.41411	95.45951	-286.108
58	71.6937	94.97449	-277.696
59	80.15685	93.39059	-268.141
60	88.05097	90.69514	-259.892
61	95.46133	86.5403	-250.195
62	102.0584	81.35092	-241.105
63	107.7669	75.1537	-232.316
64	116.205	68.25522	-231.214
65	125.1038	61.01698	-230.516
66	133.8536	53.65788	-230.563
67	142.6616	46.13229	-230.143
68	151.2337	38.83284	-229.459
69	159.6911	31.58367	-229.666
70	167.9142	24.37972	-227.786
71	172.2399	16.67175	-218.009
72	175.0321	8.458871	-208.308
73	176.5338	-0.27924	-198.311
74	176.5495	-9.06771	-188.252
75	174.8853	-17.9132	-178.5
76	171.8657	-26.0089	-169.292
77	167.3771	-33.7439	-159.402

78	161.7411	-40.4292	-149.171
79	155.5231	-48.2083	-144.692
80	149.318	-56.3546	-138.072
81	142.082	-63.9827	-135.018
82	134.2222	-71.6743	-133.51
83	126.1193	-79.1547	-133.702
84	117.9778	-86.562	-133.38
85	109.7052	-94.274	-133.175
86	101.4234	-101.592	-132.29
87	92.77633	-109.188	-131.601
88	86.95453	-116.863	-138.063
89	83.48618	-125.229	-148.282
90	81.57173	-134.139	-158.062
91	81.17023	-143.134	-168.363
92	82.28907	-151.855	-177.374
93	84.93396	-160.421	-186.087
94	88.76814	-168.366	-196.113
95	93.86991	-175.692	-206.054
96	100.1143	-181.967	-215.521
97	107.3757	-187.24	-224.912
98	115.2545	-191.032	-234.576
99	123.8472	-193.43	-243.904
100	132.5784	-194.545	-253.467
101	141.6467	-194.154	-263.082
102	150.4351	-192.259	-272.887
103	158.8578	-188.827	-282.477
104	166.4314	-183.978	-292.736

105	173.1785	-178.03	-302.57
106	178.6194	-171.345	-311.774
107	183.0743	-163.342	-321.715
108	186.0475	-154.792	-331.143
109	187.4866	-145.866	-341.477
110	187.3644	-136.953	-351.268
111	185.9335	-128.053	-360.103
112	182.9104	-119.475	-370.294
113	178.3503	-111.444	-380.515
114	172.7372	-104.466	-390.356
115	165.8253	-98.4563	-400.113
116	158.0255	-93.6998	-409.531
117	149.7054	-90.3721	-419.015
118	140.9396	-88.5139	-428.647
119	132.1086	-88.0422	-438.379
120	123.2231	-89.0322	-447.802
121	114.3244	-91.7309	-457.548
122	107.3392	-95.1639	-465.957

Table 5.6 - Reading for two circular motion on floor

References

1. https://www.researchgate.net/publication/221645389_Dead_Reckoning_for_Mobile_Robots_Using_Two_Optical_Mice?enrichId=rgreq-3843766b9040efe08de2bd72c4f78e48-XXX&enrichSource=Y292ZXJQYWdlOzIyMTY0NTM4OTtBUzoxMDIwNjMyMDc4Nzg2NTdAMTQwMTM0NTE3MzA3MQ%3D%3D&el=1_x_2&_esc=publicationCoverPdf
2. https://python-evdev.readthedocs.io/en/latest/
3. https://numpy.org/doc/
4. https://www.raspberrypi.org/
5. https://socket.io/