

Investigation Of Various Forms Of Nonlinearities In Differential Drive Wheel Mobile Robots And Its Influence On Its Kinematic Model

Thesis submitted in partial fulfilment of the requirements for the degree of
Master of Engineering in Control System Engineering

By

Sourav Sarkar

Class Roll No: 002110804014

Examination Roll No: M4CTL23006

Registration No.: 127562 of 2014-2015

Under the supervision of

Prof. Ranjit Kumar Barai

Submitted To

Department Of Electrical Engineering

Faculty Of Engineering & Technology

Jadavpur University, Kolkata – 700032

Faculty Of Engineering And Technology
Jadavpur University

Kolkata-700032

Declaration of Originality and Compliance of Academic Ethics

*I hereby declare that this thesis entitled "**Investigation Of Various Forms Of Nonlinearities In Differential Drive Wheel Mobile Robots And Its Influence On Its Kinematic Model**" contains literature survey and original research work by the undersigned candidate, as part of Master of Control System Engineering studies. All information in this document has been obtained and presented with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referred all materials and results that are not original to this thesis.*

Name : Sourav Sarkar
Class Roll Number : 002110804014
Examination Roll Number : M4CTL23006
Registration Number : 127562 of 2014-2015

Signature:

Date:

Faculty Of Engineering And Technology

Jadavpur University

Kolkata-700032

Certificate Of Recommendation

We do hereby recommend that the thesis presented under our supervision by Sourav Sarkar (Roll No: 002110804014) entitled "Investigation Of Various Forms Of Nonlinearities In Differential Drive Wheel Mobile Robots And Its Influence On Its Kinematic Model" be accepted in partial fulfilment of the requirements for the degree of Master of Engineering in Control System Engineering.

Thesis Supervisor

Prof. Ranjit Kumar Barai
Dept. of Electrical Engineering
Jadavpur University

Countersigned by

Prof. Biswanath Roy
(Head of the Department)
Electrical Engineering
Jadavpur University

Prof. Saswati Mazumdar
(Dean)
Faculty of Engineering. & Technology
Jadavpur University

Faculty Of Engineering And Technology
Jadavpur University

Kolkata-700032

Certificate Of Approval

This foregoing thesis entitled “**Investigation Of Various Forms Of Nonlinearities In Differential Drive Wheel Mobile Robots And Its Influence On Its Kinematic Model**” is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not endorse or approve any statement made, opinion expressed, or conclusion drawn there in but approves the thesis only for the purpose for which it has been submitted.

Committee for final evaluation of the thesis

Acknowledgements

I am deeply in debt and extremely thankful to my supervisor Prof. Ranjit Kumar Barai, Department of Electrical Engineering, Jadavpur University, for his noble guidance and support. He has been an ideal teacher, mentor, and thesis supervisor, offering advice and has been a true critic of my work, correcting mistakes and showing me the path whenever I encountered any problem. This thesis is a result of the countless number of one on one sessions and the late night discussions we had. His encouragement and belief in my abilities helped me complete this research work.

This project would not have been possible without the support of many people, especially my mother, Mrs. Kalpana Sarkar. She is the source of my strength and beliefs. Her presence and our fun late night coffee sessions is what helped me pull the hectic all-nighters before the deadlines. To her, I give my everything, including this. I would also like to extend my gratitude to my beloved father, Mr. Surjya Kumar sarkar, without his effort, I'm unable to reach this position.

I would also like to extend my gratitude to Mr. Arnab Mandal for assisting me throughout this project. Whenever I approached him with a new problem related to my work, he offered the best suggestions on how to tackle it effectively. His wise and valuable advice played an important role in making me possible to prepare this thesis.

I want to thank my fiancé Ms. Deblina Ghosh for being the source of my motivation. I am

indebted to her for providing a constant source of humor and inspiration for the last couple of years. Her presence greatly helped me in delivering the best possible results.

My heartfelt gratitude also goes to my friend Chirantan Gupta, who has proved to be the truest friend in all these years, standing with me through difficulties providing guidance and humor. He has also been the source of my technical clarifications and other requirements for this work.

Finally, I would also like to thank my alma mater, Jadavpur University, and all my professors for their invaluable sharing of knowledge and academic guidance which has helped me to shape myself as an enthusiastic learner of technology and as a person.

Date:

Sourav Sarkar

M.E (Control System Engineering)
Department of Electrical Engineering
Jadavpur University

Abstract

The study of wheeled mobile robot Kinematic has long been a cornerstone of robotics research, as it underpins the fundamental understanding of how these machines navigate and interact with their environments. The influence of nonlinearities in differentially powered wheeled mobile robots is a crucial issue that is often overlooked. The different nonlinearities observed in such systems are examined in this study abstract together with their significant impacts on robot Kinematic.

Differentially driven wheeled mobile robots, characterized by two independently driven wheels, offer simplicity and maneuverability, making them prevalent in various applications. However, there are major difficulties due to the intrinsic nonlinearities in their motion dynamics. Nonlinearities arise from wheel slippage, backlash in gear, variations in terrain, and mechanical imperfections, among other factors. These nonlinearities can result in deviations between the desired and actual robot trajectories.

For precise motion control, localization, and path planning, it is crucial to understand these nonlinearities. In this abstract, we discuss the common sources of nonlinearities in differentially driven wheeled mobile robots and their effects on kinematic models. We explore how these nonlinearities can lead to wheel slippage, odometry errors, and backlash in the motor gear and deviations in the robot's estimated pose.

We also explore methods for modelling and reducing these nonlinear impacts. This comprises the application of sophisticated motion planning algorithms, adaptive control techniques, and sensor fusion. By addressing these challenges, we aim to improve the overall performance and reliability of differentially driven wheeled mobile robots in real-world applications.

This abstract's conclusion emphasizes the need of taking nonlinearities into account while designing differentially powered wheeled mobile robots. Understanding and addressing these non-idealities are crucial steps toward advancing the capabilities and robustness of these robots in diverse environments, paving the way for their continued integration into various fields, such as autonomous navigation, logistics, and exploration.

Nomenclature

WMR	Wheel Mobile Robot
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
SLAM	Simultaneous Localization and Mapping
GPS	Global Positioning System
SMC	Sliding Mode Control
RTK	Real time Kinematic
ICC	Instantaneous Centre of Curvature
ω	Angular Velocity
l	Distance between two centres of two wheels
V_r	Right Wheel Velocity
V_l	Left Wheel Velocity
R	Signed distance from the ICC to the midpoint between the wheels
θ	Orientation
X	Position about X-axis
F	Radio-Frequency Identification
LED	Light Emitting Diode
SPI	Serial Peripheral Interface
I2C	Inter-Integrated Circuit
LKF	Linear Kalman Filter
H	Observation Matrix
F	State Transition Matrix
$P_{n,n}$	Covariance Update Equation
K_n	Kalman Gain
y	Position about Y-axis
\dot{X}	Velocity about X-axis

\dot{Y}	Velocity about Y-axis
$\dot{\theta}$	Angular Velocity

Contents

Acknowledgements.....	IV
Abstract	VI
Nomenclature	VIII
List of Tables.....	XIV
Chapter 1: Introduction	1
1.1 Various Kinematic configuration of WMR	1
1.1.1 Differential Drive WMR	2
1.2 Wheel Configuration for Wheel Mobile Robot (WMR).....	4
1.3 Challenges in WMR.....	6
1.3.1 Limited Mobility.....	6
1.3.2 Difficulty in Tight Space	7
1.3.3 Sensitivity to External Disturbance	8
1.3.4 Lack of Stability	8
1.3.5 Effects of Low Traction Surface.....	9
1.3.6 Limited Load Capacity.....	9
1.3.7 Maintenance and Servicing	10
1.3.8 Other Factors	11
1.4 Kinematic Stability of WMR.....	11

1.5	Scope of Work	12
1.6	Objectives.....	13
Chapter 2: Literature Review		14
2.1	Overview of Differential Drive Wheeled Mobile Robot	14
2.2	Kinematic of Differential Drive WMR	14
2.3	Dynamics of Differential Drive Wheeled Mobile Robot	15
2.4	Localization of Wheeled Mobile Robot	15
2.5	Nonlinearities in Differential Drive Wheeled Mobile Robot	16
Chapter 3: Mathematical Model of Differential Drive Kinematic.....		18
3.1	Differential Drive Kinematic.....	18
3.2	Forward Kinematic for Differential Drive Robots	19
3.3	Inverse Kinematic	21
Chapter 4: Indoor Localization: Role of Kinematic.....		22
4.1	Importance of Indoor Localization	22
4.2	Indoor Localization Methods	23
4.2.1	Sensor Fusion	23
4.2.2	Simultaneous Localization and Mapping (SLAM).....	23
4.2.3	Wi-Fi Based Localization.....	23
4.2.4	Visual Based Localization	23
4.2.5	Ultrasound and RFID	23

4.3	Challenges in Indoor Localization.....	23
4.3.1	Sensor Noise.....	24
4.3.2	Map Management.....	24
4.3.3	Computational Complexity.....	24
4.3.4	Ambiguity.....	24
4.4	Preliminary Work.....	24
4.5	Calculating Position with Wheel Encoder Data.....	27
4.6	Sensor Used in this Experiment.....	28
4.6.1	Hall Effect Magnetic Encoder.....	28
4.6.2	IMU Sensor.....	30
4.7	Actuator Specification.....	32
4.8	Controller.....	33
4.8.1	Arduino Mega 2560.....	33
4.8.2	L298 Motor Controller.....	35
4.9	Algorithm.....	36
4.9.1	Extended Kalman Filter (EKF).....	36
4.10	Plot of Trajectory of the WMR from Encoder and IMU.....	38
4.11	Nonlinearity in Measurement.....	38
Chapter 5: Physical Nonlinearities in WMR.....		40
5.1	Wheel Slip.....	40
5.1.1	Longitudinal Slip or Slippage.....	41
5.1.2	Lateral Slip or Skid.....	41

5.2	Nonholonomic Constraints.....	42
5.3	Backlash in Planetary Gear DC Motor	44
5.4	Sensor Nonlinearity	47
5.5	Sensor Nonlinearity	47
Chapter 6: Effect of Nonlinearities, Measurement Uncertainties and Noise in		
Kinematic based Odometry Physical Nonlinearities in WMR.....		48
6.1	Nonlinearities.....	48
6.1	Uncertainty in WMR.....	48
6.1	Effect of Noise in WMR	50
6.1.1	Sensor Noise.....	50
6.1.2	Acoustic Noise.....	51
6.1.3	Communication and Control Noise	51
6.1.4	Processing and Latency Noise.....	52
Chapter 7: Conclusion.....		53
Chapter 8: Future Scope.....		55
References		57
Appendix A : Arduino Code.....		62
Appendix B : Python Code		70

List of Tables

1. Table 1.1. Wheel Configuration in WMRs.	6
2. Table 4.1. Specifications of OE37.....	30

List of Figures

1. Figure 1.1. Schematic diagram of a Differential Drive Robot.....	3
2. Figure 3.1. Differential Drive Kinematic.	18
3. Figure 3.2. Forward Kinematic for Differential Drive Robot.....	20
4. Figure 4.1. The Isometric, Top, and Bottom view of the experimental setup.....	25
5. Figure 4.2. Top view along with dimensions Hall Effect Magnetic Encoder (OE37) ...	29
6. Figure 4.3. Output Wave of Hall Effect Magnetic Encoder (OE37).....	29
7. Figure 4.4. Architecture of ‘Adafruit BNO055’ IMU Sensor.....	31
8. Figure 4.5. Torque-Speed Characteristics of D.C. Motor in different Operating Voltages.	32
9. Figure 4.6. Orange Planetary Gear of D.C. Motor.	32
10. Figure 4.7. Dimensions of an Orange Planetary Gear of D.C. Motor.	33
11. Figure 4.8. Top View of an Arduino Mega 2560.	34
12. Figure 4.9. Isometric View of an L298 Motor Controller.	36

13. Figure 4.10. Trajectory of the WMR from Encoder and IMU comparing the Measured, Predicted, and Updated Data.	38
14. Figure 5.1. Pure rolling disk and its generalized coordinates in 2D plane.	43
15. Figure 5.2. Mecanum wheel can move sideways and is holonomic.....	44
16. Figure 5.3. General Classification of low backlash gearboxes.....	45
17. Figure 5.4. Planetary gearbox scheme.....	46

Chapter 1: Introduction

A wheeled mobile robot, often referred to simply as a mobile robot, is a type of autonomous or semi-autonomous robotic system designed for mobility on wheels. These robots are extensively employed for a wide range of purposes, including industrial automation, logistics, agriculture, healthcare, and even entertainment. Depending on their intended use, wheeled mobile robots can have a wide range of designs and features, but they all have the ability to move on wheels in common.

1.1 Various Kinematic configuration of WMR

The fundamental study of how mechanical systems operate is Kinematic. It's deals with the motion of objects without considering the forces causing that motion. In order to construct a mobile robot that is suitable for a task and to comprehend how to develop software for a particular instance of mobile robot hardware, it is necessary to comprehend the mechanical behaviour of the robot. There are several different Kinematic configurations for wheel robots, each with its own advantages and limitations. The choice of configuration depends on factors such as the desired manoeuvrability, stability, and terrain adaptability of the robot.

- **Differential Drive:** This configuration consists of two wheels, each driven independently by its own motor. The robot turns by driving each wheel at a different speed or direction. Differential drive robots are simple and efficient but can have difficulty with excessive wheel slippage and limited maneuverability.
- **Omni-Directional Drive:** Omni-directional robots, also known as holonomic or Mecanum robots, use wheels that are capable of both forward/backward motion and sideways strafing. By independently controlling the rotation of each wheel, these robots can move in any direction without changing their orientation. Omni-directional drive robots excel in maneuverability but can have lower traction and stability compared to other configurations.
- **Ackermann Steering:** This configuration features a set of wheels, with the front wheels steered using a mechanism known as the Ackermann steering geometry. The rear wheels are driven by a motor or motors. Ackermann steering provides precise control and good stability but is limited in maneuverability and cannot move sideways.

- **Articulated Steering:** In this configuration, the robot consists of multiple segments connected by hinges or joints. Each segment has its own wheels, and the segments can be steered independently. Articulated steering allows for high maneuverability and can adapt to uneven terrain, but it can be complex and less stable than other configurations.
- **Four-Wheel Drive:** This configuration involves each wheel being independently driven by a motor. By controlling the speed and direction of each wheel, four-wheel drive robots can achieve high traction and stability. These robots are commonly used in off-road or rough terrain applications but may have limited maneuverability.
- **Tracked Drive:** Tracked robots use continuously moving tracks instead of wheels. The tracks provide excellent traction and stability, making them suitable for challenging terrains such as sand, mud, or snow. Tracked drive robots have limited maneuverability and tend to be slower than wheeled robots.

Each kinematic configuration has its own advantages and considerations, and choosing the right configuration for a specific application requires considering the desired performance and operating conditions.

1.1.1 Differential Drive WMR

Differentially driven wheel mobile robots, often referred to as differential drive robots (Figure 1.1), are a prevalent and versatile type of mobile robot used in various applications, from industrial automation to autonomous vehicles. This essay provides a comprehensive overview of differentially driven wheel mobile robots, covering their principles of operation, advantages, limitations, and real-world applications. Differentially driven wheel mobile robots, with their simplicity, versatility, and cost-effectiveness, have earned a prominent place in the world of robotics. While they excel in certain applications, it is essential to recognize their limitations, particularly in challenging terrains. As technology advances, these robots continue to evolve, expanding their potential use cases and contributing to the automation and optimization of various industries.

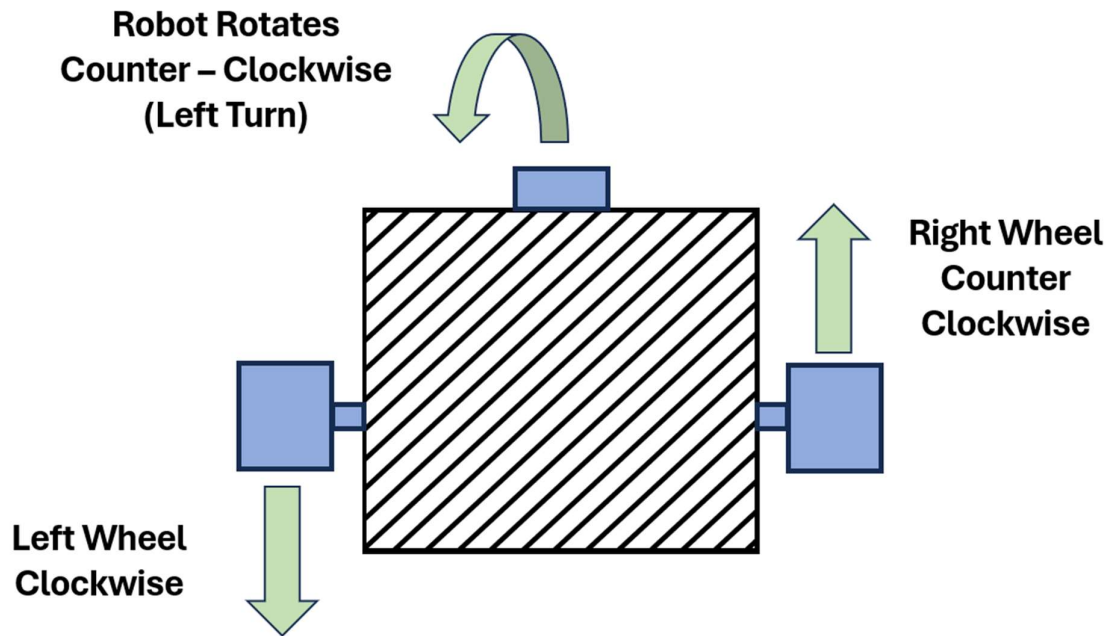


Figure 1.1. Schematic diagram of a Differential Drive Robot.

1.1.1.1 Principle of Operation

Differentially driven wheel mobile robots are characterized by their unique method of locomotion, which involves two independently powered wheels, typically located on either side of the robot. The robot's motion is controlled by varying the relative speeds of these two wheels. To move forward, both wheels rotate in the same direction at the same speed. To turn, one wheel rotates faster than the other, creating a difference in wheel speeds. This principle allows for precise control of the robot's movement, including forward and backward motion and rotation on the spot.

1.1.1.2 Advantages

- **Simplicity:** Differential drive robots are relatively simple in design, consisting of fewer moving parts compared to other locomotion mechanisms. This simplicity often results in lower manufacturing and maintenance costs.
- **Maneuverability:** These robots excel in tight spaces and can pivot in place, making them suitable for applications in environments with limited room for maneuvering.

- **Cost-Effectiveness:** Due to their simplicity, differential drive robots are cost-effective solutions for various tasks, such as material handling in warehouses and indoor navigation.
- **Versatility:** They can be equipped with a variety of sensors and payloads, allowing them to perform tasks ranging from surveillance and inspection to exploration and transportation.

1.1.1.3 Limitations

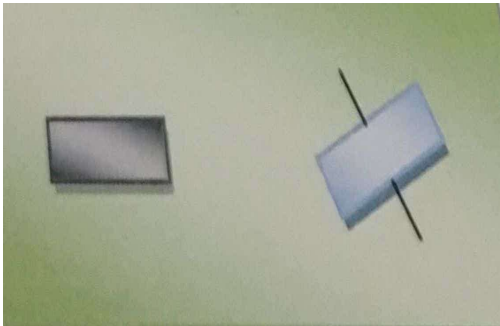
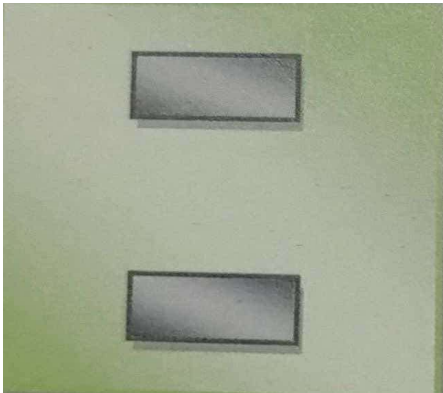
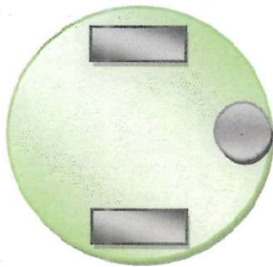
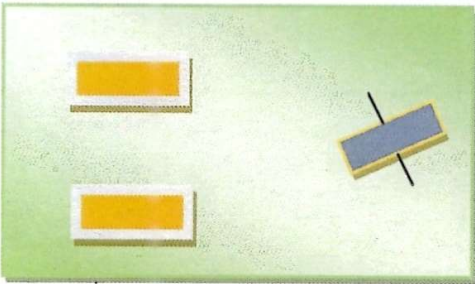
- **Limited Off-Road Capability:** Differential drive robots are less suitable for rough or uneven terrain, as they lack the stability and traction offered by more complex mobility systems like tracked or legged robots.
- **Wheel Wear:** Continuous skidding or differential wheel rotation can lead to increased wear and tear on the wheels, requiring regular maintenance.
- **Turning Challenges:** Precise turning control can be challenging, as sudden changes in wheel speeds may cause instability or even tipping in some situations.

1.1.1.4 Real-World Application

- **Manufacturing:** They are used in assembly lines for material transport, component positioning, and quality control.
- **Logistics and Warehousing:** These robots efficiently move goods within warehouses, optimizing inventory management and order fulfillment.
- **Surveillance and Security:** Differential drive robots equipped with cameras and sensors are deployed for surveillance, patrolling, and monitoring in both indoor and outdoor environments.
- **Education and Research:** They serve as platforms for teaching robotics principles and conducting experiments in educational institutions and research laboratories.
- **Agriculture:** Some agricultural robots utilize differential drive systems for tasks like weeding, planting, and crop inspection.

1.2 Wheel Configuration for Wheel Mobile Robot (WMR)

Generally, WMRs use 2 to 6 wheels for locomotion. The wheel configuration is based on the number of wheels and the geometry employed in the design of mobile robots. Table 1.1 gives a brief description of the overall configuration of wheels used in WMRs.

No. of Wheels	Configuration	Description
2		In front, 1 steering wheel, in rear, 1 traction wheel.
2		2-wheel differential-drive with the centre of mass below the axle.
2		2-wheel centred differential drive with 3 rd point of contact.
3		In rear, 2 free wheels, in front, 1steered traction wheels.

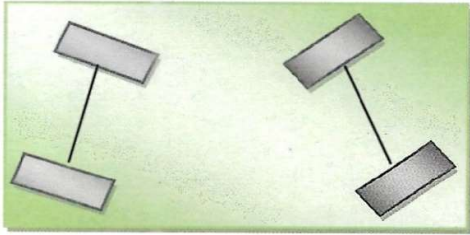
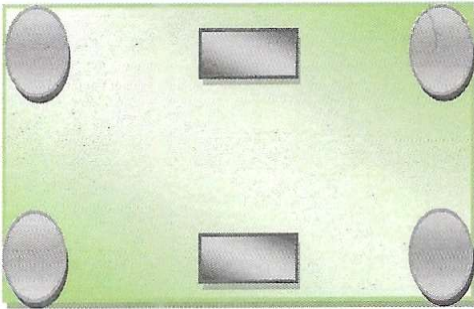
4		4 steered and motorized wheels.
6		In centre, 2 traction wheels, at the 4 corners, 1 omnidirectional wheels.

Table 1.1. Wheel Configuration in WMRs.

1.3 Challenges in WMR

1.3.1 Limited Mobility

Limited mobility is a fundamental challenge for wheel mobile robots. It refers to their inability to navigate certain terrains or overcome obstacles that pose difficulties for their wheeled locomotion system. This limitation is primarily due to the design and operation of the wheels, which may struggle with rough or uneven surfaces, steep inclines, and obstacles such as stairs or curbs. One of the main factors affecting the mobility of wheel mobile robots is the traction between the wheels and the ground. Traction is critical for enabling the wheels to generate sufficient force and maintain grip on the surface to propel the robot forward. However, low-traction surfaces, such as ice or wet floors, can significantly reduce the effectiveness of the wheels and impede their ability to move.

Furthermore, the wheel design also plays a role in determining the level of mobility. For example, conventional wheels with a cylindrical shape may struggle with traversing rough terrains due to their limited contact area with the ground. This can lead to reduced stability and an increased risk of the robot tipping or falling over. Uneven or unstable surfaces pose additional mobility challenges for wheel mobile robots. The inability of the wheels to adapt to changes in terrain can cause uneven weight distribution and disturbance to the stability of the

robot. As a result, the robot may struggle to maintain its balance and may become immobilized or exhibit erratic behaviour.

Tight spaces can also present mobility difficulties for wheel mobile robots. Their wheelbase, the distance between the centres of the front and rear wheels, influences their ability to maneuver in narrow areas. If the wheelbase is wide, the robot may find it challenging to navigate through tight passages or make sharp turns. In summary, the limited mobility of wheel mobile robots stems from factors such as traction limitations, wheel design, stability issues on uneven surfaces, and constraints in tight spaces. Overcoming these challenges requires innovative solutions in wheel design, suspension systems, and locomotion control algorithms to enhance the mobility of wheel mobile robots and allow them to navigate diverse terrains and environments.

1.3.2 Difficulty in Tight Space

Wheel mobile robots can face difficulty in movement in tight spaces due to several reasons:

- **Kinematic constraints:** The wheel configuration and dimensions of the mobile robot can limit its ability to turn or navigate through tight spaces. For example, a robot with a long wheelbase may find it difficult to make sharp turns.
- **Wheel slippage:** In tight spaces, the wheels of a mobile robot may lose traction and slip, making it difficult to control the robot's movement accurately.
- **Obstacle avoidance:** In tight spaces, there may be various obstacles like walls or furniture that can impede the movement of the robot. The robot's sensing and navigation systems must be able to detect and navigate around these obstacles.
- **Turning radius:** The turning radius of a mobile robot determines how sharply it can turn. In tight spaces, a large turning radius can make it challenging to change direction or maneuver around obstacles.
- **Control algorithms:** The control algorithms of the robot must be able to accurately control the wheel speeds and steering angles to navigate through tight spaces. Inaccurate control can lead to collisions or getting stuck.

To overcome these challenges, some possible solutions include implementing differential or omnidirectional wheel configurations, using sensors for accurate obstacle detection and

avoidance, and employing sophisticated control algorithms for precise and reliable movement in tight spaces.

1.3.3 Sensitivity to External Disturbance

Wheel mobile robots can be sensitive to external disturbances due to several factors:

- **Wheel traction:** If the wheels of a mobile robot do not have sufficient traction, they can easily slip or lose control when encountering external disturbances. This can affect the robot's ability to navigate accurately.
- **Centre of gravity:** The location of the robot's center of gravity can affect its stability and resistance to external disturbances. If the center of gravity is not properly balanced, the robot may become unsteady or tip over when subjected to external forces.
- **Wheel suspension:** The suspension system of the wheels can also play a role in the robot's sensitivity to external disturbances. If the suspension is not adequately designed to absorb shocks or vibrations, the robot can be easily affected by uneven terrain or disturbances in its environment.
- **Control system:** The control algorithms and feedback systems used in the robot's control system can also impact its sensitivity to external disturbances. If the control system is not designed to handle disturbances robustly, the robot may have difficulty maintaining stability and accurate movement.

Some ways to mitigate the sensitivity to external disturbances include improving the traction of the wheels, optimizing the centre of gravity to enhance stability, implementing a robust suspension system, and designing a control system that can adapt to external disturbances. Additionally, using sensor feedback to detect and respond to disturbances in real-time can help the robot adjust its movements accordingly.

1.3.4 Lack of Stability

One of the main factors contributing to the lack of stability of a wheel mobile robot is its high centre of gravity. This means that the mass of the robot is concentrated towards the top, making it more prone to tipping over. Additionally, the design and arrangement of the wheels can affect the stability of the robot. If the wheels are located too far apart, it can increase the risk of tipping. Similarly, if the wheels are too small, it can make the robot more susceptible to getting stuck or losing balance on uneven surfaces.

Another factor that can affect the stability of a wheel mobile robot is the speed at which it is moving. Higher speeds can lead to more unstable movements, especially when the robot needs to make quick turns or maneuvers. To improve the stability of a wheel mobile robot, various measures can be taken. These include lowering the centre of gravity by redistributing the mass and using a wider base for better balance. Additionally, implementing algorithms and control systems that account for the dynamic behaviour of the robot can help improve stability during movement.

1.3.5 Effects of Low Traction Surface

A low traction surface can have a significant impact on the performance and movement of a wheel mobile robot. When the wheels of the robot do not have enough grip on the surface, it can lead to slipping, sliding, or even total loss of control.

One of the main challenges with low traction surfaces is the reduced friction between the wheels and the ground. This can make it more difficult for the wheels to generate enough force to propel the robot forward or change its direction. As a result, the robot may struggle to move or maintain steady movement and can even get stuck in certain situations.

Another issue with low traction surfaces is that they can cause the wheels to spin or skid. This can occur when the driving force applied to the wheels exceeds the available friction between the wheels and the ground. In such cases, the wheels might rotate without effectively transmitting the motion to the robot, resulting in inefficient or erratic movements.

To address these challenges, there are several strategies that can be implemented. One approach is to use wheels with specialized treads or materials that are designed to improve traction on low friction surfaces. Another option is to adjust the control algorithms of the robot to account for the reduced traction and optimize the motion accordingly. Additionally, adding additional weight or using stabilizing mechanisms can help improve the stability and grip of the robot on low traction surfaces.

1.3.6 Limited Load Capacity

The limited load capacity of a wheel mobile robot refers to its inability to carry heavy loads or handle excessive weight. This constraint can impact the robot's functionality and limit its application in certain scenarios.

The load capacity of a wheel mobile robot depends on various factors, such as the strength of the robot's structure, the power of its motors, and the design of its wheels. If any of these components are not adequately designed or configured to handle heavy loads, the robot may struggle to carry the weight, resulting in reduced performance or potential damage to the robot.

In some cases, the limited load capacity of a wheel mobile robot can also be a safety concern. If the robot is required to transport or manipulate heavy objects, exceeding its load capacity can lead to unstable movements, increased risk of tipping over, or even structural failure. To overcome the limitations of load capacity, several strategies can be employed. One approach is to reinforce the structure of the robot and use stronger materials to increase its overall strength. Another option is to upgrade the motors and drive systems to generate more power and torque, enabling the robot to handle heavier loads. Additionally, using larger wheels or implementing mechanisms like suspension systems can distribute the weight more evenly and improve the robot's load carrying capabilities.

1.3.7 Maintenance and Servicing

Wheel mobile robots require regular maintenance and servicing, particularly for their wheels and suspension systems, to ensure proper functionality and prevent premature wear and tear.

- **Component failures:** Wheel mobile robots consist of various components such as motors, wheels, sensors, and controllers. These components can fail over time due to wear and tear or manufacturing defects. Identifying and replacing faulty components can be a time-consuming and costly process.
- **Wear and tear on wheels:** The wheels of a wheel mobile robot are subjected to constant friction and load during operation, which can cause them to wear out. Regular maintenance and replacement of wheels are necessary to ensure optimal performance and prevent damage to the robot and the surface it operates on.
- **Sensor calibration and alignment:** Most wheel mobile robots rely on sensors such as cameras or LiDAR to perceive their surroundings and navigate through the environment. These sensors need to be calibrated and aligned correctly to provide accurate readings and avoid errors in perception. Sensor calibration can be a complex and time-consuming task, requiring specialized knowledge and equipment.
- **Battery maintenance:** Wheel mobile robots are often powered by rechargeable batteries. Regular inspection, charging, and replacement of batteries are necessary to ensure

consistent and reliable operation. Battery maintenance also involves monitoring battery health, preventing overcharging or discharging, and ensuring proper storage and disposal of old batteries.

- **Software updates and troubleshooting:** Wheel mobile robots are typically controlled by software programs that provide instructions for navigation, obstacle avoidance, and other tasks. Regular software updates are necessary to improve performance, fix bugs, and add new features. Troubleshooting software-related issues can be challenging and may require expertise in programming and robotics.
- **Environmental conditions:** Wheel mobile robots are often used in harsh environments such as outdoor construction sites, warehouses, or factories. Exposure to dust, moisture, extreme temperatures, and other environmental factors can damage the robot's components and affect its performance. Regular maintenance and protection measures, such as sealing or shielding sensitive components, are necessary to ensure longevity and reliability.

1.3.8 Other Factors

Power consumption: Wheel mobile robots often require a significant amount of power to operate, especially when traversing challenging terrains. This can limit their battery life and overall endurance.

Cost: Depending on the complexity and features of the wheel mobile robot, they can be relatively expensive to design, develop, and manufacture. This cost may limit their accessibility and adoption in certain industries or applications.

Noise and vibration: Wheel mobile robots can produce noise and vibration while in operation, which may be undesirable in certain environments, such as hospitals or quiet workspaces

1.4 Kinematic Stability of WMR

The Kinematic stability of a wheel mobile robot refers to its ability to maintain stability and balance while in motion. this is particularly important for wheeled robots that operate on uneven or unstable surfaces. One common issue that can affect the Kinematic stability of a wheel mobile robot is wheel slippage. when the wheels of the robot slip or lose traction, it can cause the robot to lose stability and potentially tip over. slippage can occur when the surface is

slippery, when the wheels are worn, or if the robot is moving too fast or trying to make sharp turns. To improve Kinematic stability, several factors need to be considered:

- **Wheel design:** the design of the wheels and their materials can greatly impact stability. wheels with good traction, larger contact patches, and a lower coefficient of friction are preferred to prevent slippage.
- **Wheel diameter and width:** increasing the diameter and width of the wheels can improve stability since it increases the contact area with the ground and distributes the robot's weight more evenly.
- **Centre of gravity:** the distribution of weight in the robot affects its stability. placing heavy components lower to the ground can help lower the center of gravity, reducing the chances of tipping or instability.
- **Suspension system:** a well-designed suspension system can help absorb shocks and vibrations, allowing the wheels to maintain contact with the ground and improving stability.
- **Control algorithms:** implementing proper control algorithms that consider the robot's Kinematic and environment can help prevent instability. this includes adjusting wheel speeds, acceleration rates, and turning angles to mitigate slippage and maintain stability.
- **External sensors:** using sensors such as accelerometers or gyroscopes can help detect any changes in the robot's orientation or stability and trigger appropriate adjustments in the control system.

Regular maintenance and monitoring of the wheel mobile robot's components and their performance are also essential for ensuring Kinematic stability. this includes inspecting and replacing worn wheels, checking for proper wheel alignment, and ensuring that the control software is adequately calibrated.

1.5 Scope of Work

When looking at past papers, most of the work till date centred on indoor localisation using many algorithms and using many sensors, effect of non-holonomic constraints on the dynamics and Kinematic of the wheeled mobile robot and also the nonlinearities like systematic and non-systematic errors in the sensor readings and process of calibration. But the effect of backlash in the Kinematic of WMR has played a vital role. Due to this backlash on the gear of the DC motor, for a forward and reverse movement of a WMR, both the encoder displays different

readings. So, the odometry defined for this model contains errors. To remove this model, we have proposed a model by which we can define odometry in a different way so that the backlash and the other nonlinearity may be removed from the WMR and we'll get a precise error free odometry.

1.6 Objectives

This study approaches with the following objectives.

1. This chapter serves as an introduction to the thesis, providing an overview of the research topic and the scope of the work.
2. In this chapter, the focus is on the latest literature survey regarding the topics of WMR.
3. In this chapter, the focus is on mathematical model of the forward and inverse Kinematic of the Differential Drive WMR.
4. This chapter focusses on the experimental part of the research of Indoor localization of WMR. Analyse and discuss the collected data, particularly related to encoder and IMU chip.
5. It focuses on the different types of Nonlinearities working on the WMR.
6. It focuses on the effects of Nonlinearities on the Kinematic model of the WMR.

Chapter 2: Literature Review

2.1 Overview of Differential Drive Wheeled Mobile Robot

A differential drive wheeled mobile robot employs two independently driven wheels positioned symmetrically on its chassis to govern motion [1]. Each wheel is given its own amount of power, which makes turning and orienting modifications easier. These robots are widely used because of their efficient and streamlined design, which improves mobility and compactness. By connecting wheel velocities to rotational and translational behaviours, the mathematical framework of differential Kinematic controls the motion dynamics of these robots. Because of their versatility and user-friendly control, they have numerous uses in exploration, interior navigation, and instructional robotics. In-depth studies and literature explore their kinematic characteristics, control mechanisms, and practical applications [2-5].

2.2 Kinematic of Differential Drive WMR

Literature investigates the Kinematic of differential drive mobile robots, including motion analysis and control [6]. Design, control methods, kinematic modelling, and motor analysis are important topics of research [7]. Research focuses on the use of wheel encoders for precise rotation and the kinematic analysis of wheeled robots traversing difficult terrain [2]. For differential drive robots, research also includes motion planning, control algorithm creation, and behavioural architecture exploration [6,7]. Comparative examination of differential and omnidirectional drive systems reveals the relative advantages and disadvantages of each [8]. Additionally, the improvement of non-holonomic differential drive robot control is aided by modelling and experimental investigations [5]. A strong foundation for understanding the kinematic behaviour and control systems of differential drive wheel mobile robots has been laid forth by this thorough literature survey.

In two studies, the Kinematic of a mobile differential drive robot is thoroughly investigated. Malu and Majumdar's initial study offers dual trajectory planning techniques. One method involves turning the robot in order to correct an orientation problem, and then translating in order to correct a distance error. Another approach addresses both orientation and distance problems at once by combining rotational and translational motion. By incorporating robot movement, localization can be determined, and the control law is based on a Kinematic model that updates the reference speed for high-frequency PID control of the DC motor. The

Lyapunov Criterion confirms that the control law is stable [9,10]. The motion planning and control of a robot with differential drive are covered in detail in Kothandaraman's second article. It includes work such as kinematic equation derivation and proportional controller implementation. It covers kinematic equation derivation, proportional controller implementation, leader-follower, pursuer-evader, obstacle detection, forms, and point-to-point motion. For waypoint assignment, many robots are used, and a GUI software is created. The article incorporates numerous data processing and communication systems and technologies [11,12].

2.3 Dynamics of Differential Drive Wheeled Mobile Robot

There is a wealth of academic literature on the study of differential drive wheel mobile robots. Notably, a study project presented an adaptive sliding mode control strategy based on the robot's dynamic model, with the purpose of improving steering stability and accuracy [13]. A third piece of research developed a tri-wheel layout for a differential drive mobile robot that uses two wheels for differential propulsion and one universal wheel for steering and equilibrium [14]. The conceptualization and control of a rimless wheel robot with differential drive was the subject of a separate investigation. This robot enabled straight-line locomotion and turning through carefully controlled motor current modulation, which resulted in differential current addition and subtraction [15]. Significantly, a study that developed a sliding mode controller skilfully drove the sliding surface and error to a convergence at zero, boosting the trajectory pursuit capabilities of the differential drive mobile robot [4] studied the field of trajectory tracking control. Last but not least, a thorough investigation examined the subtleties of mobile robots pulling off-axle trailers under various two-wheel configurations in an effort to improve the integrated system's operating capability and effectiveness in the actual world [13]. The combination of these academic endeavours offers a thorough understanding of the developments and approaches influencing the dynamics and control of mobile robots with differential drive wheels.

2.4 Localization of Wheeled Mobile Robot

Localization of wheeled mobile robots in indoor environments is challenging due to noise and obstacles. To forecast robot coordinates and cut down on noise, people frequently use the extended Kalman filter (EKF)[16]. Another method for improving position accuracy and addressing problems like wheel-slip involves combining data from sensors like wheel

encoders, IMU, and optical mice with an EKF [17,18]. Geometric restrictions are used to indirectly combine fixation visual data for "soft sensor" localization [19]. For GPS-denied situations, a multi-sensor fusion approach that takes use of IMU rotation and wheel odometry linearity combines ORB-SLAM, IMU, and wheel odometry [14]. Using edge detection and a particle filter technique, an indoor positioning system incorporates motion sensor and visual feature data to estimate poses precisely [19].

2.5 Nonlinearities in Differential Drive Wheeled Mobile Robot

Different approaches are used to deal with non-linearity in mobile robots using differential drives. Trajectory tracking and error convergence are both successfully achieved using sliding mode control (SMC) [20,21]. Control accuracy and stability are improved by a dynamic surface control rule with a nonlinear gain function [22,23]. Neural networks help to improve steering stability and accuracy by approximating uncertain robot model sections [24,25]. These techniques reduce non-linearities, enhancing the performance of mobile robots with differential drives [25,26].

Researchers around the world have done considerable work on modelling of wheel slip & skid in WMRs. Wang and Low presented a WMR model in the presence of wheel skidding and slippage from the perspective of control design [27]. In mid-nineties, Balakrishna and Ghosal [28] modelled slip in a WMR considering it to be non-holonomic assuming that wheel undergoes no pure-rolling in practical circumstances. Both of these scientists were also involved in presenting a GPS based skidding and slippage control for a car-like robot. The robot incorporated Real-Time Kinematic (RTK)-GPS and other sensors to measure the WMR's posture, velocities, and perturbations due to wheel skidding and slipping for control compensation [29]. Just recently, Ozoemenae et al. [30] modelled and analysed a five wheeled robot with slip on an uneven terrain.

Scientific community reports significant research on slip & skid control for WMRs. Kececi and Tao [31] addressed the problem of stability and tracking of a vehicle during slippage of its wheels without braking using an adaptive vehicle skid control. In 2010, Amodeo et al. [32] developed a sliding mode traction controller to control the wheel slip during skid braking and spin acceleration of a WMR. S. J. Yoo [33] proposed an adaptive tracking control of a WMR encountering with unknown skidding and slipping. A path tracking control of a WMR considering slippage and unknown dynamics was implemented by W. Dong [34]. In recent

times, Magallanet al. proposed a sliding-mode observer to estimate the wheel slip and vehicle velocity under unknown road conditions by measuring only the wheel speed [35]. Afterwards Tian and Sarkar have presented a study of multi-robot formation control subject to slip [36]. Lately in 2013, Ćirović and Aleksendrić have been involved in presenting a couple of research works involving an adaptive fuzzy logic for slip control [37] and neural network based longitudinal slip control [38]. Considering all of the aforementioned research works, it is evident that slip and skid control has been very hot topic among roboticists. The present thesis is an effort to extend this research by focusing slip and skid control on slippery terrains.

Chapter 3: Mathematical Model of Differential Drive Kinematic

Kinematic is the most basic study of how mechanical systems behave. In mobile robotics, we need to understand the mechanical behaviour of the robot both in order to design appropriate mobile robots for tasks and to understand how to create control software for an instance of mobile robot hardware.

In our work we're using a differentially drive wheeled mobile robot. So, we're deriving the forward and inverse Kinematic for a differentially drive mobile robot.

3.1 Differential Drive Kinematic

Many mobile robots use a drive mechanism known as differential drive. It consists of two drive wheels mounted on a common axis, and each wheel can independently be driven either forward or backward [3].

While we can vary the velocity of each wheel, for the robot to perform rolling motion, the robot must rotate about a point that lies along their common left and right wheel axis. The point that the robot rotates about is known as the ICC - Instantaneous Centre of Curvature (see Figure 3.1).

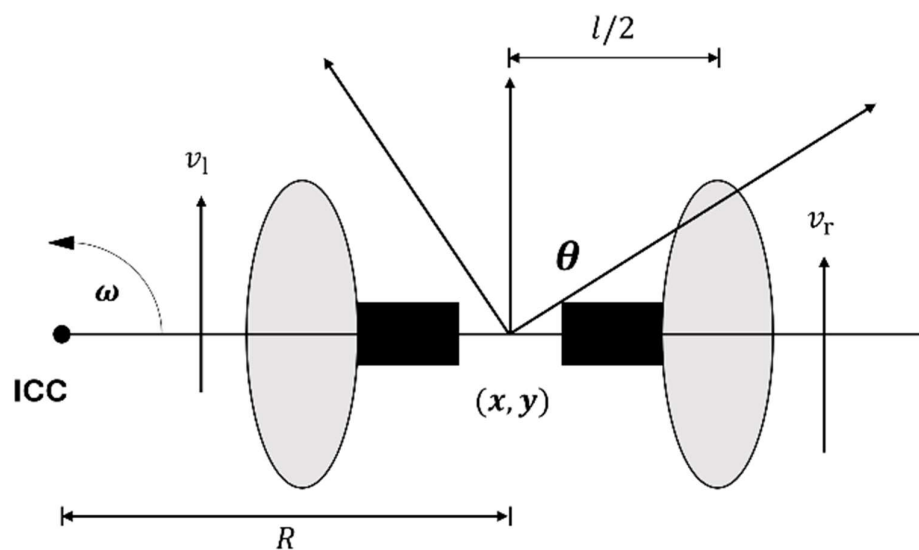


Figure 3.1. Differential Drive Kinematic.

By varying the velocities of the two wheels, we can vary the trajectories that the robot takes. Because the rate of rotation ω about the ICC must be the same for both wheels, we can write the following equations 3.1 and 3.2:

$$\omega \left(R + \frac{l}{2} \right) = v_r \quad 3.1$$

$$\omega \left(R - \frac{l}{2} \right) = v_l \quad 3.2$$

Where l is the distance between the centres of the two wheels v_r , v_l are the right and left wheel velocities along the ground, and R is the signed distance from the ICC to the midpoint between the wheels. At any instance in time, we can solve for R and ω using Eqn. 3.3.

$$R = \frac{l}{2} \frac{v_l + v_r}{v_r - v_l}; \quad \omega = \frac{v_r - v_l}{l} \quad 3.3$$

There are three interesting cases with these kinds of drives.

- If $v_l = v_r$, then we have forward linear motion in a straight line. R becomes infinite, and there is effectively no rotation - ω is zero.
- If $v_l = -v_r$, then $R = 0$, and we have rotation about the midpoint of the wheel axis we rotate in place.
- If $v_l = 0$, we have rotation about the left wheel. In this case $R = \frac{l}{2}$, same is true if $v_r = 0$.

Note that a differential drive robot cannot move in the direction along the axis - this is a singularity. Differential drive vehicles are very sensitive to slight changes in velocity in each of the wheels. Small errors in the relative velocities between the wheels can affect the robot trajectory. They are also very sensitive to small variations in the ground plane and may need extra wheels (castor wheels) for support.

3.2 Forward Kinematic for Differential Drive Robots

In figure 1.1, assume the robot is at some position (x, y) , headed in a direction making an angle θ with the X axis. We assume the robot is centered at a point midway along the wheel axle. By

manipulating the control parameters v_l , v_r , we can get the robot to move to different positions and orientations. (note: v_l , v_r) are wheel velocities along the ground). Knowing velocities v_l , v_r and using equation 3.3, we can find the ICC location using Eqn. 3.4.

$$\text{ICC} = [x - R \sin \theta, y + R \cos \theta] \quad 3.4$$

and at time $(t + \delta t)$ the robot's pose will be given by Eqn. 3.5.

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega \delta t) & -\sin(\omega \delta t) & 0 \\ \sin(\omega \delta t) & \cos(\omega \delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - \text{ICC}_x \\ y - \text{ICC}_y \\ \theta \end{bmatrix} + \begin{bmatrix} \text{ICC}_x \\ \text{ICC}_y \\ \omega \delta t \end{bmatrix} \quad 3.5$$

This equation simply describes the motion of a robot rotating a distance R about its ICC with an angular velocity of ω as shown in Figure 3.2.

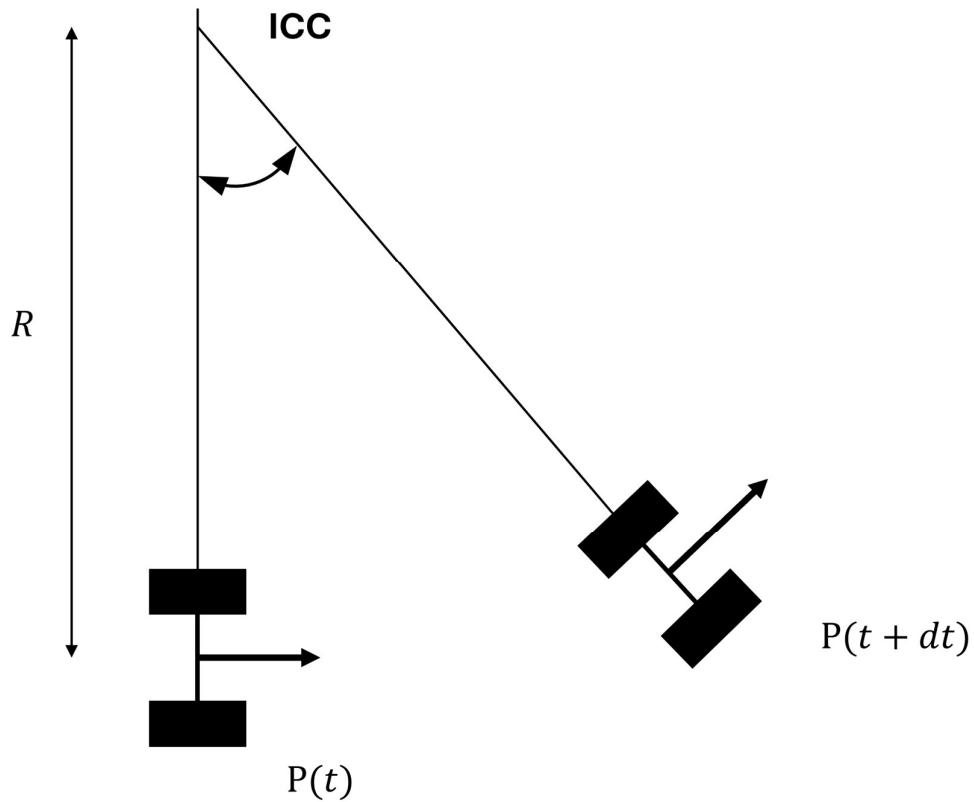


Figure 3.2. Forward Kinematic for Differential Drive Robot.

3.3 Inverse Kinematic

In general, we can describe the position of a robot capable of moving in a particular direction $\theta(t)$ at a given velocity $V(t)$ using Eqns. 3.6, 3.7, and 3.8.

$$x(t) = \int_0^t V(t) \cos[\theta(t)] dt \quad 3.6$$

$$y(t) = \int_0^t V(t) \sin[\theta(t)] dt \quad 3.7$$

$$\theta(t) = \int_0^t \omega(t) dt \quad 3.8$$

A related question is: How can we control the robot to reach a given configuration (x, y, θ) - this is known as the inverse Kinematic problem [3].

Unfortunately, a differential drive robot imposes what are called non-holonomic constraints on establishing its position. For example, the robot cannot move laterally along its axle. A similar nonholonomic constraint is a car that can only turn its front wheels. It cannot move directly sidewise, as parallel parking a car requires a more complicated set of steering maneuvers. So, we cannot simply specify an arbitrary robot pose (x, y, θ) and find the velocities that will get us there.

Chapter 4: Indoor Localization: Role of Kinematic

Wheeled mobile robots have various applications in the field of industry, military, and security environment. The problem of autonomous motion planning and control of wheeled mobile robots have attracted lot of research interest in the field of robotics. Consequently, engineers working on design of mobile robots have proposed various drive mechanisms to drive such robots. However, the most common way to build a mobile robot is to use two-wheel drive with differential steering and a free balancing wheel (castor). Controlling the two motors independently make such robots to have good manoeuvring and work well in indoor environment. Mobile robots with such drive systems are a typical example of non-holonomic mechanisms due to the perfect rolling constraints on a wheel motion (no longitudinal or lateral slipping).

In case of Outdoor localization, GPS played a vital role, but accuracy of the GPS is a questionable thing, so in that with the help of encoder and data fusion techniques, we can define a better outdoor localization but in case of Indoor localization, GPS is not worked properly. So, in that case with the help of different sensors, we can define a good localization.

Wheeled mobile robots (WMRs), which are employed in a variety of industries, including manufacturing, logistics, healthcare, and even smart homes, depend heavily on indoor localization. It entails figuring out the robot's location and orientation inside of a building without frequently using external equipment like GPS. For wheeled mobile robots, this introduction gives a general understanding of indoor localisation.

4.1 Importance of Indoor Localization

WMRs need indoor localisation for a number of reasons:

- WMRs frequently operate in limited or dynamic interior areas where accurate positioning is required for safe and effective navigation. Robots can avoid obstacles, create the best courses, and execute jobs precisely with the help of accurate localization.
- WMRs are used in many areas, including as autonomous material handling in warehouses and medical aid in hospitals, to complete activities without the need for human involvement. For the robot to function dependably and securely in such settings, accurate localization is essential.

- In order to map the environment accurately or successfully explore unknown areas, WMRs need to know their precise position and orientation.

4.2 Indoor Localization Methods

4.2.1 Sensor Fusion

To determine their location, WMRs frequently use a mix of sensors, including wheel encoders and inertial measurement units (IMUs). To increase accuracy, sensor data is combined using techniques like Kalman filters or particle filters.

4.2.2 Simultaneous Localization and Mapping (SLAM)

SLAM algorithms allow robots to simultaneously build maps of their environment while localizing themselves within it. This is particularly useful when operating in unknown or changing environments.

4.2.3 Wi-Fi Based Localization

Utilizing Wi-Fi signals and access points, WMRs can estimate their position based on signal strength and proximity to known Wi-Fi access points. Although inexpensive, this procedure may not be as accurate in all circumstances.

4.2.4 Visual Based Localization

Cameras and computer vision techniques can be used to recognize and track visual markers or features in the environment, enabling localization. This method is especially effective in well-lit environments.

4.2.5 Ultrasound and RFID

Ultrasound sensors and Radio-Frequency Identification (RFID) tags can be used for localization in specific applications, such as indoor navigation for the visually impaired.

4.3 Challenges in Indoor Localization

Indoor localization for WMRs presents several challenges:

4.3.1 Sensor Noise

It can be difficult to sustain precise localization over time since noise, drift, and ambient influences can all affect sensors.

4.3.2 Map Management

Updating and maintaining precise maps of interior settings, particularly those that are dynamic, can be time- and resource-consuming.

4.3.3 Computational Complexity

Some localization methods, like SLAM, can be computationally intensive and may require powerful hardware.

4.3.4 Ambiguity

Robots may find it difficult to accurately pinpoint their location in congested or featureless settings, which can cause localization ambiguity.

4.4 Preliminary Work

Indoor localization is a fundamental challenge in the field of robotics, enabling wheeled mobile robots (WMRs) to operate autonomously in various environments. This research work explores a novel approach to indoor localization using a combination of Hall Effect magnetic encoders and Inertial Measurement Unit (IMU) sensors. The use of these sensors improves the autonomy and navigational skills of WMRs by enabling reliable and precise location within interior areas. This study presents an overview of the methodology, experimental setup, and results, showcasing the potential of this hybrid sensor fusion technique for indoor robot localization.

Indoor localization is a critical component of autonomous wheeled mobile robots, enabling them to navigate and perform tasks in confined spaces such as manufacturing plants, warehouses, hospitals, and smart homes. The precision and dependability of conventional systems, such as wheel encoders and IMU sensors, are constrained. This study investigates how Hall Effect magnetic encoders and IMU sensors might work together to overcome these drawbacks and offer WMRs a more reliable indoor localization solution. The setup is shown in Figure 4.1.

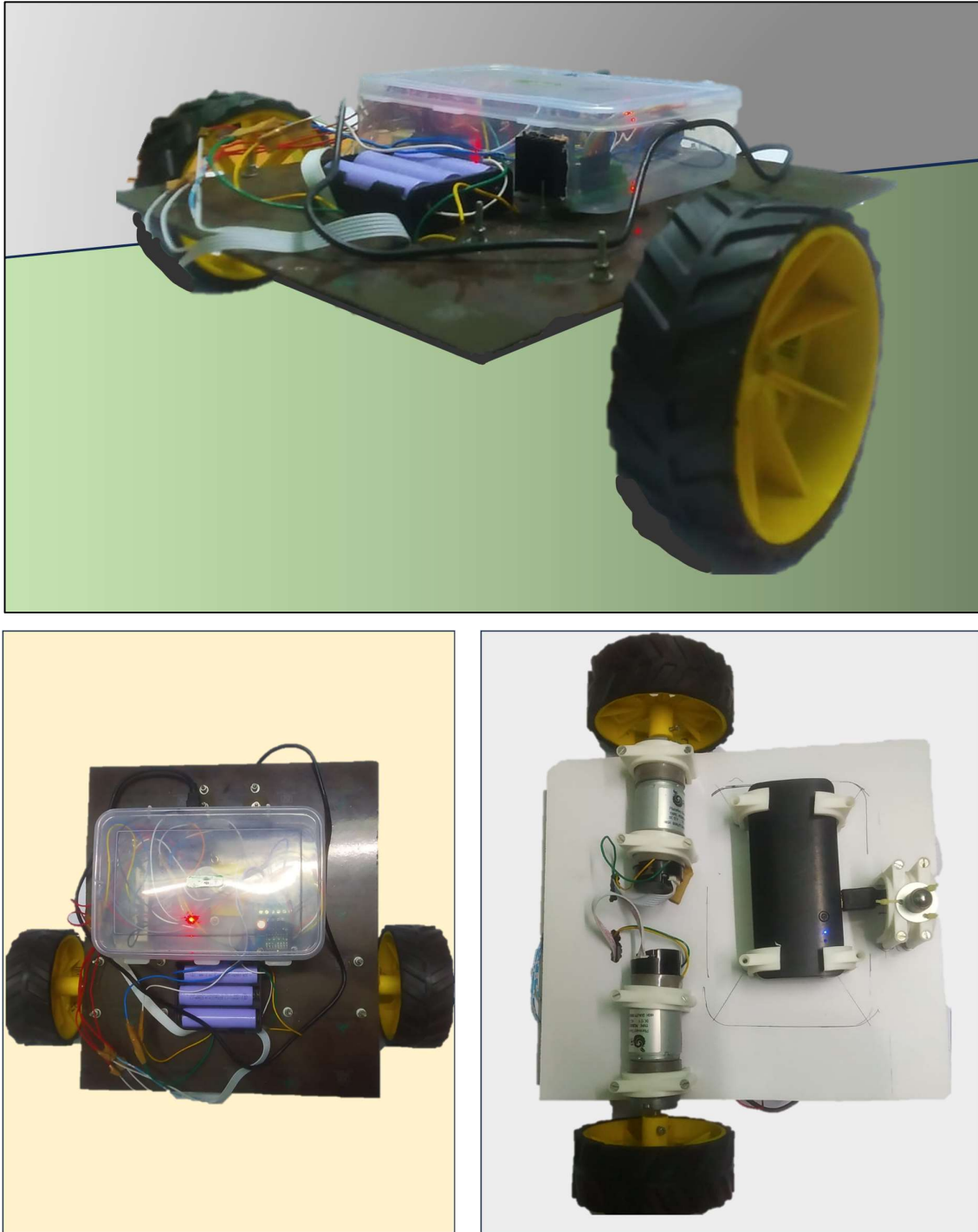


Figure 4.1. The Isometric, Top, and Bottom view of the experimental setup.

We used Hall Effect Magnetic encoders instead of Optical encoder due to some reasons:

Optical encoders use a light source (typically an LED) and a photodetector (usually a photodiode or phototransistor) to create a pattern of alternating light and dark regions on a

rotating disk or code wheel. As the disk or code wheel rotates, the photodetector detects changes in light intensity, allowing the encoder to determine position and motion. Where Hall effect magnetic encoders utilize the Hall effect, which is the generation of a voltage difference (Hall voltage) in a conductor or semiconductor when subjected to a magnetic field. Hall effect sensors (often Hall effect ICs) detect changes in the magnetic field to identify position and motion in these encoders, which typically feature a magnetic strip with alternating North and South poles.

Even the life span of optical encoder are less due to wear and tear of the optical components, particularly if used in high speed and high precision applications and these encoders are sensitive to environmental factors like dust, dirt, moisture, and ambient lightning conditions. But for hall effect magnetic encoders, are often more durable and have a longer operational life because they have no physical contact parts that can wear out. This makes them suitable for high-reliability applications and are more robust in harsh environments. They are less susceptible to dust, dirt, and moisture since they are typically sealed, making them suitable for applications with exposure to these elements.

Hall effect magnetic encoder are lightly weight rather than the optical ones and can measure position with directions but in case of Optical ones, only position can be determined.

After that the data extracted from the encoder which is mainly the number of steps i.e., the wheel velocity is merged with the data extracted from the IMU sensor, and with the help of data fusion techniques like Extended Kalman Filter [EKF], we can estimate the position of the differentially driven wheeled mobile robot from a reference coordinates.

In our case, the result is not up to the mark due to effect of nonlinearities like Wheel slip, Wheel deformation and mainly Backlash.

So, we're proposing some methods for future work for this topic to eliminate the Nonlinearities from the system and get a more accurate indoor localization of the wheeled mobile robot.

4.5 Calculating Position with Wheel Encoder Data

Two motors are identical in nature and the encoders associated with the motors, are also identical in nature.

For a complete 360° rotation of a wheel, $2\pi r$ distance will be covered.

Let's say for a complete rotation of 360° , the number of tick counts = n.

$$2\pi r \text{ cm} = n \text{ ticks} \quad 4.1$$

$$1 \text{ cm} = \frac{n}{2\pi r} \text{ ticks} \quad 4.2$$

$$L \text{ cm} = \frac{n \times l}{2\pi r} \text{ ticks} \quad 4.3$$

Let's take an example of wheel encoder data,

Left wheel encoder value= 284.

Right wheel encoder value=280.

Angle turned (Orientation angle) = 2° .

According to model of the robot:

Ticks per revolution = 84.

Wheel circumference = 18.888

Calculation Part:

Average ticks = (Left Motor Encoder Value + Right Motor Encoder Value) / 2 = 282

Distance travelled = Average ticks * Circumference

X= Distance travelled * $\sin(\text{Angle turn})$

Y= Distance travelled * $\cos(\text{Angle turn})$

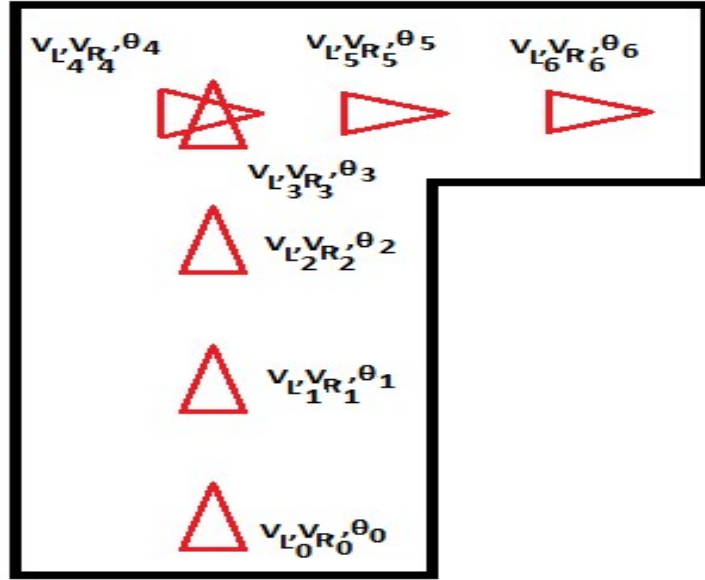


Figure 4.2. Trajectory Model of a Wheel Mobile Robot

4.6 Sensor Used in this Experiment

In our work, we mainly use two sensors for localization purpose. Hall Effect magnetic encoder and Adafruit BNO055 (IMU) is used for localization purpose and Bluetooth module HC05 is mainly used for the communication purpose.

4.6.1 Hall Effect Magnetic Encoder

A Hall Effect magnetic encoder is a sensor that operates based on the Hall Effect, a fundamental physical phenomenon. It is used to measure the position, rotation, or proximity of a magnetic source. A Hall Effect magnetic encoder's operation can be described as follows:

The Hall Effect is a phenomenon that occurs in conductive or semi conductive materials when subjected to a magnetic field. A voltage that is perpendicular to both the current and magnetic field directions is produced when a current-carrying conductor or semiconductor is subjected to a magnetic field that is parallel to the direction of current flow. This voltage is known as the Hall voltage.

The Hall Effect Sensor is a semiconductor device that includes a tiny strip of material through which current flows. Normally, the Hall Effect sensor is positioned close to the magnetic source.

Power Source: An external voltage source provides power to the Hall Effect sensor. In our case we take Arduino's 5 volts supply pin as an external voltage source.

The sensor's output as shown in Figure 4. is processed and turned into a useful signal by the output circuitry. Amplifiers, signal processing, and digital or analog output circuits could be included.

Position Calculation: Using the processed signal, the position or motion of the magnetic source relative to the sensor can be calculated. The specific method of position calculation depends on the encoder's design and application.

In our case we're using 'OE37' (see Figure 4.) as a Hall Effect Encoder which is mainly a two-channel encoder. The specifications are mentioned in Table 4.1.

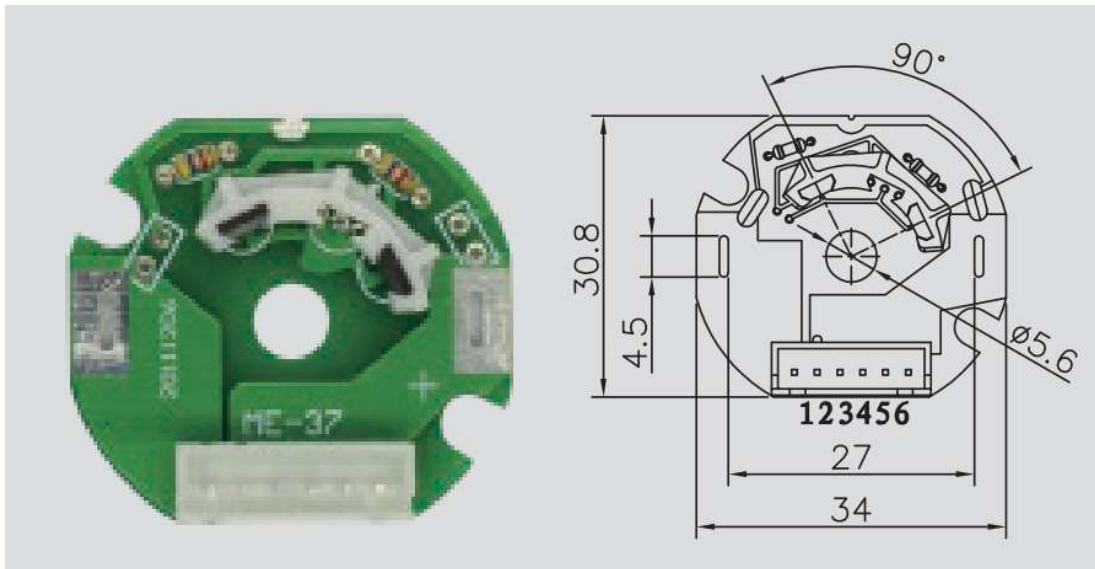


Figure 4.3. Top view along with the dimensions of Hall Effect Magnetic Encoder (OE37) [39].

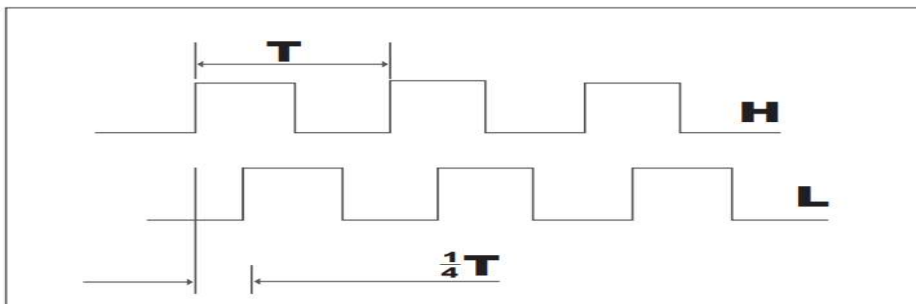


Figure 4.4. Output Wave of Hall Effect Magnetic Encoder (OE37) [39].

Characteristics	Symbol	Test Conditions	Min	Ref	Max	Units
Supply Voltage	V_{cc}	-----	4.5	--	24	V
Supply Current	I_{ce}	-----	--	14	20	mA
Output Current	I_c	$V_{ce}=12V$; Gauss<170	--	< 0.1	20	mA
Output Leakage Current	I_{cex}	Output Open; 25°C	--	--	10	μA
Output Rise Time	T_r	RL=820 Ω ; CL=20 pf; 25°C	--	0.5	1.5	μS
Output Fall Time	T_f	RL=820 Ω ; CL=20 pf; 25°C	--	0.2	1.5	μS

Table 4.1. Specifications of OE37 [39].

4.6.2 IMU Sensor

IMU sensors, or inertial measurement units, are essential parts of modern technology. They act as adaptable instruments for a variety of applications, from wearable technology and consumer electronics to aircraft and robotics. An IMU sensor is an essential piece of equipment that makes it possible to detect a variety of physical characteristics, such as acceleration and angular velocity, and to figure out how an item is oriented in three dimensions. The importance of IMU sensors depends in its capacity to change industries and improve user experiences by offering crucial data for navigation, motion tracking, stabilization, and control systems. Typically, an IMU sensor consists of the following essential parts:

- **Accelerometers:** These sensors are used to calculate changes in velocity by measuring linear acceleration along various axes.
- **Gyroscopes:** Gyroscopes calculate the rate of rotation around particular axes by measuring angular velocity.
- **Magnetometers:** Magnetometers are used in some IMU systems to measure the earth's magnetic field and provide orientation estimates in relation to magnetic north.
- **Microcontroller Unit (MCU):** An MCU performs computations and sensor fusion to analyse input from accelerometers, gyroscopes, and other sensors to determine direction and motion.

Sensor fusion, which includes merging data from many sensors to produce a more precise picture of an object's location, orientation, and motion, is an essential feature of IMU sensors. This procedure improves the accuracy and precision of IMU-based measurements by making up for the inherent limits and faults of individual sensors. Here, ‘Adafruit BNO055’ was used which is a 9-axis absolute orientation (see Figure 4.).

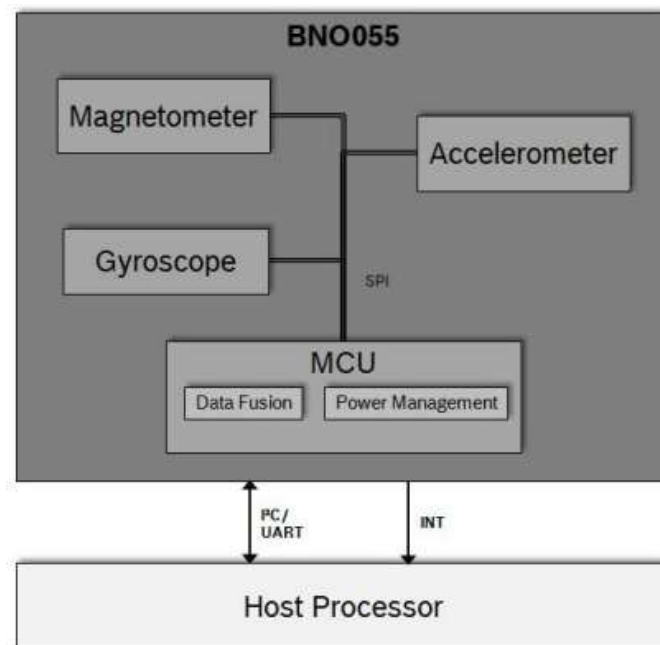


Figure 4.5. Architecture of ‘Adafruit BNO055’ IMU Sensor [41].

4.7 Actuator Specification

In this work, we used Orange Planetary Gear DC Motor (Figure 4. and Figure 4.) which is a 12V DC planetary motor which has a gearbox of 35 mm diameter. The Torque-Speed Characteristics of the motor is shown in Figure 4.. The planetary type of gearbox of this motor has a 26.9: 1 reduction ratio which produces 185 RPM with the torque value of around 72.5 N-cm.

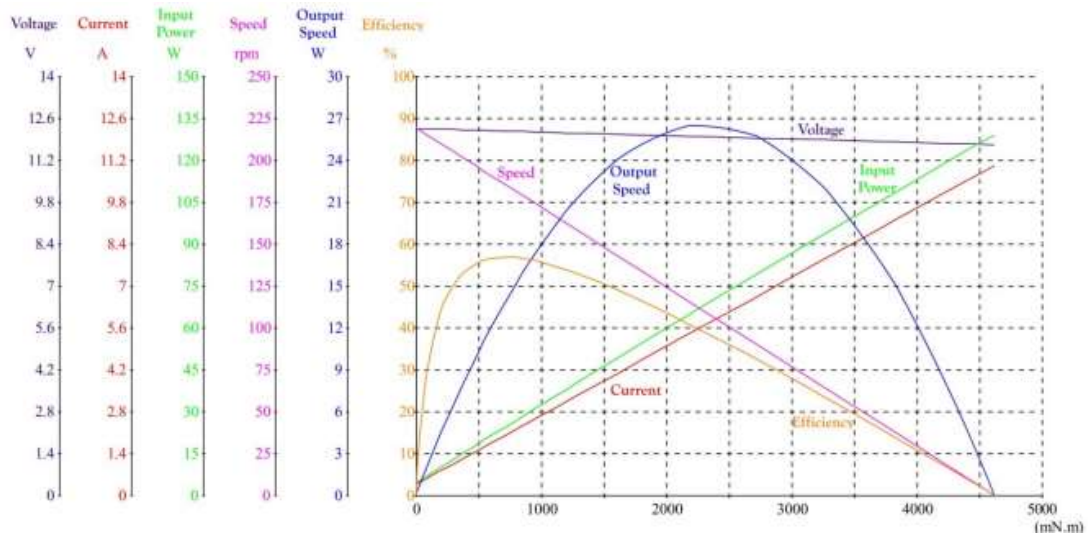


Figure 4.6. Torque-Speed Characteristics of D.C. Motor in different Operating Voltages [40].



Figure 4.7. Orange Planetary Gear of D.C. Motor [40].

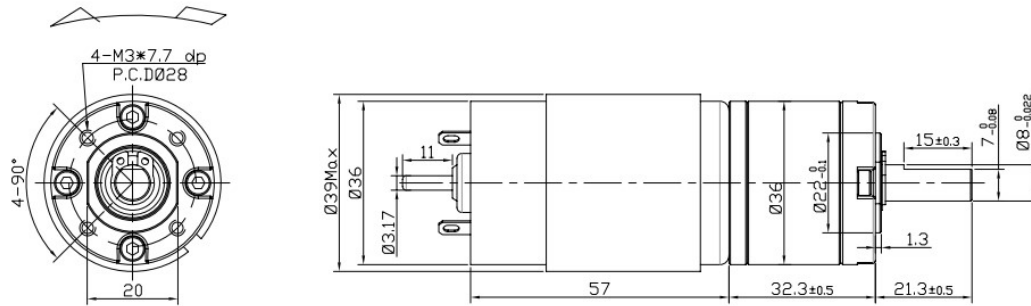


Figure 4.8. Dimensions of an Orange Planetary Gear of D.C. Motor [40].

4.8 Controller

A controller is a device that continually monitors a system's output (or state) and compares it to a desired or reference value (set point). It is frequently implemented in software or hardware. It then computes and applies control actions to the system to minimize the error between the actual output and the desired output, thereby maintaining the system's performance within specified limits.

We used Arduino Mega 2560 as a main controller in our work and L298 H-bridge motor controller for controlling the velocity of the two Planetary Gear DC Motor.

4.8.1 Arduino Mega 2560

The Arduino Mega 2560 is a popular microcontroller board that is part of the Arduino ecosystem. It's based on the ATmega2560 microcontroller and is known for its expanded capabilities compared to other Arduino boards, making it suitable for more complex and demanding projects. An overview of the Arduino Mega 2560 is given below:

Microcontroller: The ATmega2560, an 8-bit AVR microcontroller created by Atmel (now a division of Microchip Technology), is the brains of the Arduino Mega 2560. It features 256 KB of flash memory for storing programs, 8 KB of SRAM, and 4 KB of EEPROM, and runs at a clock frequency of 16 MHz.

Digital and Analog Pins: The Arduino Mega 2560 features a total of 54 digital input/output pins, 15 of which can be used as PWM (Pulse Width Modulation) output pins. For reading analog signals, it also includes 16 analog input pins.

Communication Interfaces: The board is equipped with various communication interfaces, including:

- **USB:** To enable programming and serial connection, the Arduino Mega 2560 may be linked to a computer by USB.
- **UART (Universal Asynchronous Receiver-Transmitter):** It has four hardware UART serial communication ports for connecting to other devices or modules.
- **SPI (Serial Peripheral Interface):** SPI pins for connecting to SPI devices such as sensors, displays, and memory.
- **I2C (Inter-Integrated Circuit):** I2C pins for syncing with OLED screens and other I2C-compatible hardware.
- **CAN (Controller Area Network):** A CAN communication controller for CAN networks interface, widely used in industrial and automotive applications.

External Interrupts: The board supports external interrupts on a number of digital pins, allowing it to respond to external events in real-time.

Power Source: The Arduino Mega 2560 may be supplied by either an external DC power supply (7–12V) or a USB connection. It contains a voltage regulator that can power different components with both 5V and 3.3V.

Compatibility: The Arduino Mega 2560 (see Figure 4.) is programmable using the Arduino programming language, which is a condensed form of C/C++, and is compatible with the Arduino IDE (Integrated Development Environment).



Figure 4.9. Top View of an Arduino Mega 2560.

4.8.2 L298 Motor Controller

The L298 Motor Controller as shown in Figure 4., is a popular integrated circuit (IC) used to control and drive DC motors and stepper motors. It's commonly employed in robotics, mechatronics, and various electronic projects to control the speed and direction of motors. The L298 IC is excellent for a variety of motor control applications since it can tolerate reasonably high current and voltage. An overview of the L298 motor controller is provided below:

Key characteristics and details:

Motor Driver: The L298 IC can operate two DC motors or one stepper motor because it is essentially a dual H-bridge motor driver. Each H-bridge has the ability to speed-control a motor and operate it in both forward and backward directions.

Operating Voltage: The L298 motor controller is suited for a variety of motors since it normally runs within a voltage range of 4.5V to 46V.

Current Handling: It can handle peak currents of up to 2A per channel (with proper heat sinking) and continuous currents of about 1A per channel.

Control Signals: The L298 requires four digital input signals for each motor (two for direction control and two for speed control). You may do this to regulate each motor's direction and speed separately.

Built-in Diodes: The IC has built-in fly back diodes (also known as freewheeling diodes), which help protect against voltage spikes generated by the motor during switching.

Pinout: Although the L298 IC contains several pins, the following are some of the crucial pins:

- IN1 and IN2: These pins control the direction of motor 1.
- IN3 and IN4: These pins control the direction of motor 2.
- ENA and ENB: These pins control the speed (PWM signal) of motors 1 and 2, respectively.
- OUT1, OUT2, OUT3, OUT4: These pins are connected to the terminals that drive the motors.

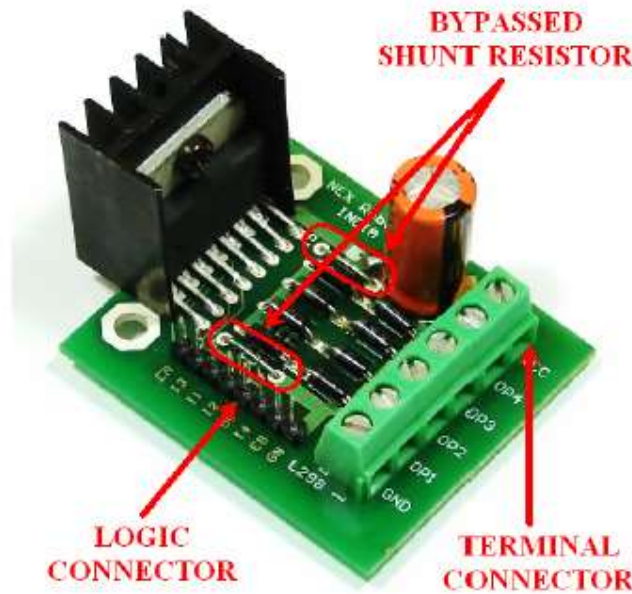


Figure 4.10. Isometric View of an L298 Motor Controller.

4.9 Algorithm

An algorithm is a detailed process or collection of instructions for completing a given activity or addressing a particular issue. An essential idea in computer science, algorithms have a wide range of uses, from data processing and sorting to intricate computations and decision-making.

For Indoor Localization of the Differential drive Wheeled Mobile Robot, We used Data Fusion algorithm mainly. Due to Nonlinear Kinematic of the mobile robot, we mainly used Extended Kalman Filter (EKF). The main idea behind the EKF is a linearization of the dynamic model at the working.

4.9.1 Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) is a widely used estimation technique in robotics and control systems. It is particularly useful when dealing with nonlinear dynamic systems. EKF extends the basic Kalman Filter by approximating the system dynamics and measurement functions using linearization techniques such as Taylor series expansion. This allows it to estimate the state of a non-linear system, making it suitable for tasks like robot trajectory estimation.

The reason we should not use a Linear Kalman Filter for robot trajectory estimation is that it assumes linearity in both the system dynamics and measurement model. In reality, many robotic systems exhibit non-linear behaviour, making the Linear Kalman Filter inadequate for accurate state estimation. EKF, on the other hand, can handle these nonlinearities by linearizing the model at each time step, providing a more accurate trajectory estimation for complex robotic systems.

The concept of the EKF is similar to the LKF (Linear Kalman Filter); however, some modifications should be made. The modifications are related to the observation matrix H and the state transition matrix F .

If the state-to-measurement relation (the first type of non-linearity) of the system is non-linear, the observation matrix is of the type:

$$H = h(x_n) \quad 4.4$$

The State update equation looks like the following:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n(Z_n - h(\hat{x}_{n,n-1})) \quad 4.5$$

For the uncertainty propagation, the observation matrix H should be linearized to keep the uncertainty PDF Gaussian. The Covariance Update Equation looks like the following Eqn. 4.6.

$$P_{n,n} = \left(I - K_n \frac{\partial h}{\partial x} \right) P_{n,n-1} \left(I - K_n \frac{\partial h}{\partial x} \right)^T + K_n R_n K_n^T \quad 4.6$$

The Kalman Gain Equation is given by Eqn. 4.7.

$$K_n = P_{n,n-1} \frac{\partial h^T}{\partial x} \left(\frac{\partial h}{\partial x} P_{n,n-1} \frac{\partial h^T}{\partial x} + R_n \right)^{-1} \quad 4.7$$

4.10 Plot of Trajectory of the WMR from Encoder and IMU

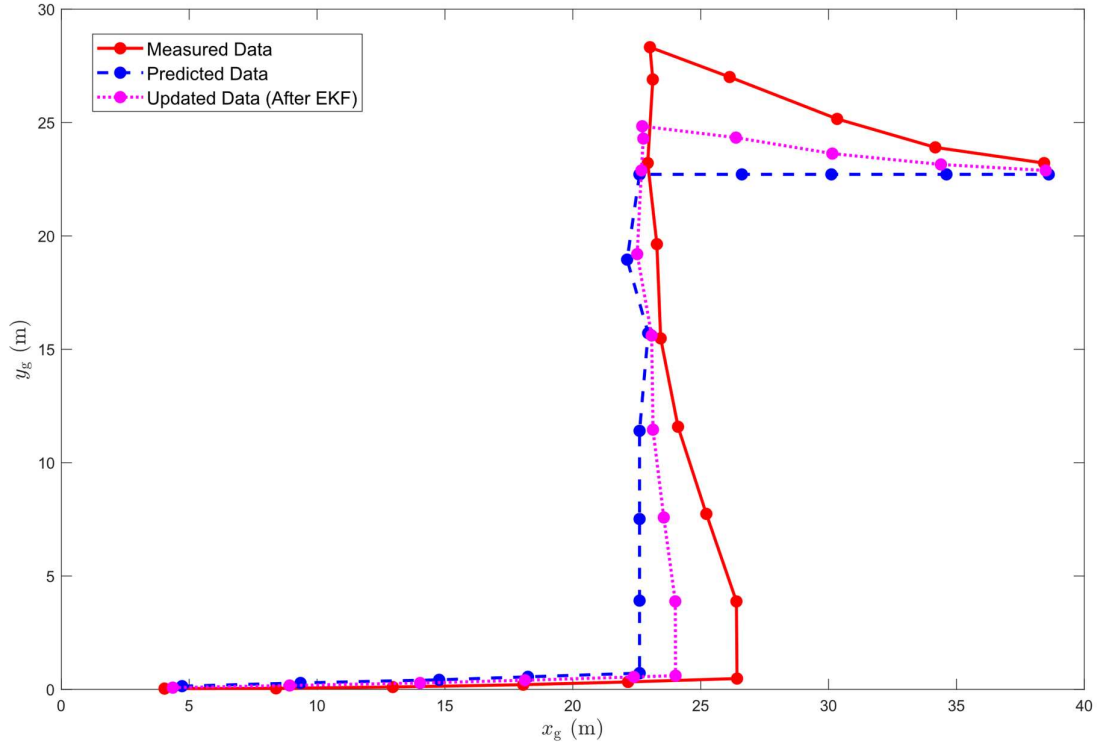


Figure 4.11. Trajectory of the WMR from Encoder and IMU comparing the Measured, Predicted, and Updated Data.

In this plot (Figure 4.), the trajectory of the differentially driven wheeled mobile robot is plotted. Where the Red line is mainly the measured data from IMU and Hall Effect magnetic encoder, the predicted one is plotted by the Blue lines and the updated plot which is derived from the EKF, is the updated trajectory.

Here, we can use different types of numerical methods like Lagrangian Interpolation for validation of the position.

4.11 Nonlinearity in Measurement

Here we're using two uniform motors of same characteristics. However when we're trying to measure the current and resistance of each motor, we have got different readings from this motors.

From this observation we can say that both the motors have not same characteristics and for a same voltage, the speed of the motors are not same. So, this refers nonlinearity in the system. Thus we have use PWM signal to control the speed of these actuators intended to mitigate the nonlinearity and improve the performance of the overall system.

When we're trying to measure the wheel diameter of the robot, the diameter of the right wheel is 111.86 mm while the diameter of the left one is 112.38 mm. It means whether the two motors are rotate the same amount of rotation, but due to different radius of the wheels both cannot cover same distance.

So, if the diameter of both the wheels and the speed of each motor are not same, then we can say that the robot model is nonlinear one. To localize this nonlinear robot model, we need a better control action and algorithms, which can measure the error and update it to the controller for a better navigation and localization.

Chapter 5: Physical Nonlinearities in WMR

Wheeled mobile robots (WMR) play a crucial role in various applications, from warehouse automation to planetary exploration. These robots offer manoeuvrability and simplicity in design, primarily due to their differential drive system, where two independently controlled wheels enable movement. However, this ostensibly simple design incorporates nonlinearities that have a substantial influence on the mobility and control of the robot.

Wheel slip in mobile robot, is a common cause of non-linearity and is mostly noticeable on rough and uneven surfaces. Another key contributor to its nonlinear behaviour is the kinematic model governing the relationship between the wheel velocities and its motion. This model is nonlinear because the robot's round wheel motion requires the use of trigonometric functions in the kinematic equations.

Wheel encoders, which are frequently employed for localization and odometry, may contribute inaccuracies and nonlinearities. These mistakes may be caused by issues including encoder resolution restrictions, wheel slippage, and mechanical flaws.

5.1 Wheel Slip

Wheel slip, which is most noticeable on uneven or slippery surfaces, is a notable source of non-linearity. The robot moves differently than what would have been expected by the control commands. As a result, one of the wheel slides, and the other maintains traction. Wheel slippage can be caused by a number of things, including frictional differences, surface imperfections, and wheel-terrain interactions.

A wheeled mobile robot's real position might deviate significantly from its ideal position due to slippage between the wheels and the ground. This particular aberration is made worse when the encoder data are utilized to control the wheel's angular position, which is then used to control the mobile robot's Cartesian position and orientation. Slip is unavoidable with differential drive robots because the left and right wheels must move at different speeds to establish the mobile robot's alignment in the plane.

Depending on its direction, slip has two types; longitudinal and lateral slippage. 'Longitudinal slippage' is also referred to as 'slippage' while 'lateral slip' is also called 'skidding'.

5.1.1 Longitudinal Slip or Slippage

Longitudinal slip happens when the speed of motion of the vehicle does not remain equal to wheel's linear velocity [13]. We call it longitudinal slippage because it is parallel to the direction of motion of robot. It is referred simply slippage in common understanding thus in this thesis we will use slippage for longitudinal slippage. One of the main reasons of this type of slip is tire deformation, where the speed of tire decreases due to its deformation, but inertia of vehicle wants the vehicle to move on an increased speed hence forcing it wheels to slip.

5.1.2 Lateral Slip or Skid

This type of slip is perpendicular to the direction of motion of wheel hence called lateral slippage or simply skid. While negotiating a sharp turn a cornering force acts on the wheel of the robot displacing it away from its plane. This movement of wheel away from its plane is called skid.

Eliminating slippage and skid in wheel mobile robots is critical for precise control and consistent performance. Here are some tactics and things to think about:

- **Tire Design:** The right tire choice and design are crucial. Slippage can be decreased by specialized tires with strong traction characteristics. The size, kind of rubber, and tread design of the tires should all be appropriate for the robot's intended environment.
- **Control Algorithms:** Use advanced control algorithms that can quickly identify and correct slippage. These algorithms can change the direction and speed of the wheels during operation to reduce or completely get rid of slippage.
- **Sensor Fusion:** Integrate data from many sensors, such as wheel encoders, accelerometers, and gyroscopes, to precisely estimate the robot's motion and detect slippage. Making quick control adjustments is made easier with sensor fusion.
- **Terrain Analysis:** Use on board sensors and cameras to analyse the terrain. Adaptive control strategies can then be employed based on the perceived surface conditions to prevent slippage.
- **Machine learning:** Use machine learning strategies to control predicted slippage. The robot can predict probable slippage scenarios and proactively change its actions by learning from previous encounters.
- **Feedback Control:** To maintain stability and minimize slippage, use feedback control

loops that continuously track the robot's movement and modify wheel speeds.

- **Simulation and modelling:** Create precise simulations of the robot's behaviour, including its slipping characteristics. Simulate different scenarios to fine-tune control strategies and reduce slippage in real-world situations.
- **Wheel Material and Surface:** To optimize friction and minimize sliding on a variety of surfaces, experiment with various wheel materials and surface treatments.

By combining these tactics and employing modern control algorithms, wheel mobile robots may significantly minimize slippage and skid, enabling more precise and reliable operations in a wide range of settings and applications.

5.2 Nonholonomic Constraints

Wheel mobile robots with nonholonomic constraints have motion restrictions that prevent them from moving freely in all directions. Nonholonomic robots have limitations on their ability to move because of the design of their wheels, in contrast to holonomic systems, which may move in any direction without restrictions.

Differential drive systems, which have two independently controlled wheels, are common in wheel mobile robots. These systems have nonholonomic restrictions, chiefly due to the fact that they are unable to move laterally without rotating.

First, holonomic and non-holonomic systems have to be defined. Let's consider a mechanical system with generalized coordinate's $q \in C$, where C is the configuration space of the proposed system and coincides with R^n . For such system, a constraint is called Kinematic when it only involves generalized coordinates (q) and velocities (\dot{q}).

Kinematic Constraints are usually defined in Pfaffian Form given by Eqn. 5.1.

$$v_i^T(q) \dot{q} = 0 \quad [i = 1, \dots, k < n] \quad 5.1$$

Where v_i 's are k linearly independent vectors. If all of the kinematic constraints defined by Eqn. 5.1 are integrable to a form of Eqn. 5.2.

$$h_i(q) = m_i \quad [i = 1, 2, \dots, k < n] \quad 5.2$$

Where, m_i is the integration constant, then they are considered to be holonomic constraints and the system subjected to them is called a holonomic system [14]. Joints in a robotic manipulator are common example of such constraints.

Each holonomic constraint causes a loss of accessibility of the system in its configuration space. Hence, for a system with k holonomic constraints, the accessible configurations are reduced to a $(n - k)$ dimensional subset of C .

A non-holonomic system on the other hand, is subjected to at least one non integrable (i.e. non-holonomic) constraint. Although such constraint limits the local mobility of the system, due to its non-integrable nature, the accessibility to C is not affected. Hence, generalized coordinates are not reduced. However, generalized velocities in a system subjected to k non-holonomic constraint belongs to a $(n - k)$ dimensional subspace.

Wheels are typical sources of non-holonomic constraints. Consider the disk in Figure 5.1 with generalized coordinates $q = [x \ y \ \theta]^T$, assuming the disk can only roll on the touching plane without slipping to the sides (i.e., there is no velocity component for the contact point perpendicular to the plane containing the disk).

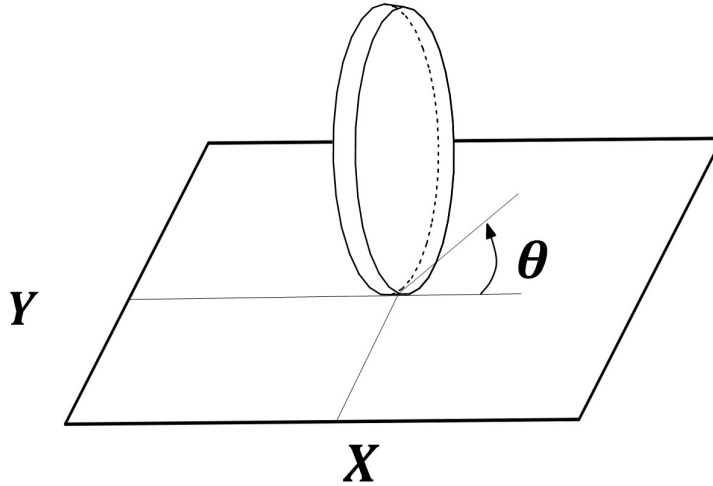


Figure 5.1. Pure rolling disk and its generalized coordinates in 2D plane.

This can be defined using Eqn. 5.3.

$$\dot{x} \sin \theta - \dot{y} \cos \theta = 0 \quad 5.3$$

Rewriting Eqn. 5.2 in Pfaffian form will result in Eqn. 5.4.

$$[\sin \theta \quad -\cos \theta \quad 0]\dot{q} = 0 \quad 5.4$$

As it can be seen, Equation 1.3 is not integrable causing the nature of the wheel to be non-holonomic. Also, it should be emphasized that this constraint implies no loss in accessibility of the wheel configuration space, meaning that wheel can reach any goal $q_f = [x_f \ y_f \ \theta_f]^T$ starting from any initial state $q_i = [x_i \ y_i \ \theta_i]^T$.



Figure 5.2. Mecanum wheel can move sideways and is holonomic.

This kinematic constraint applies to all wheel-based systems, making them non holonomic. However, it should be noted that not all wheels are non-holonomic. Configuration of caster wheel proposed in mic or Mecanum wheels (as shown in Figure 5.2), which are commonly used in omnidirectional robots, are exempt from this constraints and in fact are considered, holonomic.

5.3 Backlash in Planetary Gear DC Motor

Often in the context of gears or links, backlash is a word used in engineering and other fields to describe a phenomena where there is a delay or play in the movement of mechanical components. An introduction to the idea is provided below:

Backlash, in a mechanical system, is the space between two mating parts or the lost motion between them. It shows how much motion one part can make without also moving the other part right away.

Backlash is caused by tolerances, clearances, or gaps in the mechanical parts. It may be brought on by flaws in the manufacturing process, normal wear and tear, or deliberate design decisions made to avoid binding.

Backlash, which is measured in angular minutes (arc min), is the rotation angle of the output shaft when the input shaft is stationary. The gearbox is deemed to have little backlash if the value of this angle is less than 10 arc min [4]. The backlash between the gear teeth and the backlash in the ball bearings are the two most important factors that affect the overall amount of backlash in toothed gearboxes. It is possible to completely remove their impact on the overall backlash of the toothed gearbox by selecting an appropriate type of bearing.

The value of backlash is not very significant for universal toothed gearboxes [5], but it is significant for low backlash gearboxes (see Figure 5.3) because low backlash between meshing gears is required for the requisite positioning accuracy.

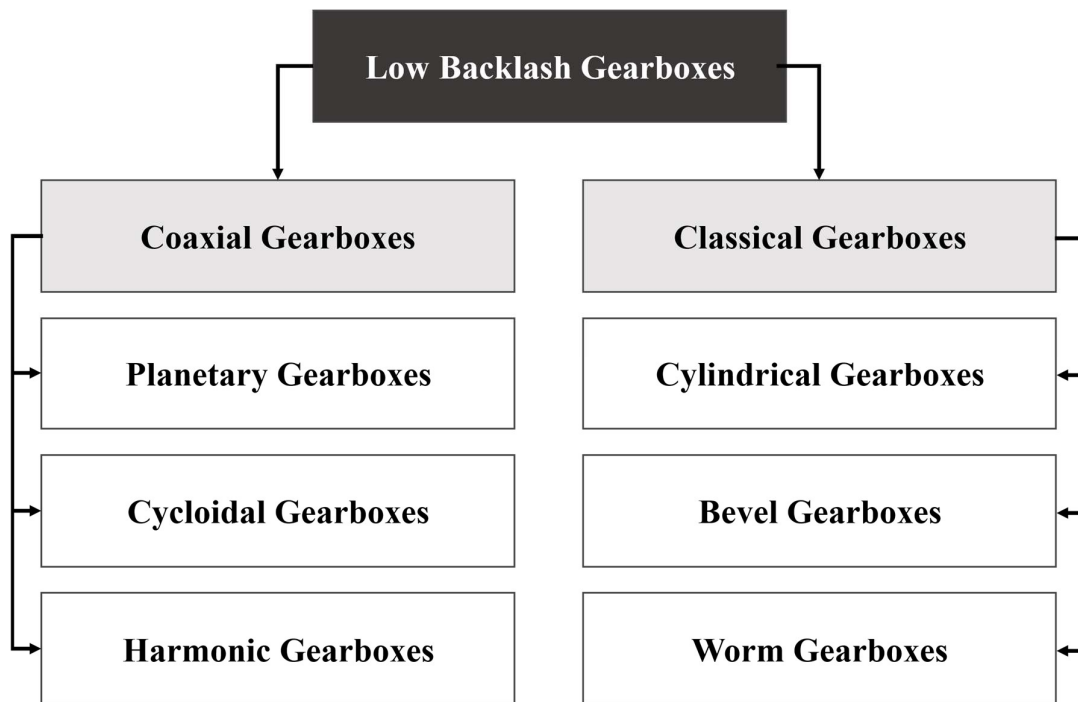


Figure 5.3. General Classification of low backlash gearboxes.

The most popular planetary gearboxes (see Figure 5.4) are those with spur gears and low backlash because they are tiny, straightforward, and inexpensive and have little arc backlash. They are produced by the Neugart firm, with a transmission ratio of 3...10 (single-stage), an

arc backlash of 1 arc min (micro-accuracy), 3 arc min (enhanced accuracy), and 6 arc min (standard accuracy), and an efficiency coefficient of 97...98% [6].

Additionally, helical gear planetary gearboxes with low backlash are utilized frequently. They are straightforward but cost more than the earlier models and have a smaller backlash because of the stiffer gears.

They are produced by the Neugart business with a coefficient of efficiency of 98%, a transmission ratio of 4...10 (single-stage), an arc backlash of 1 arc min (micro-precision), and 3 arc min (standard accuracy).

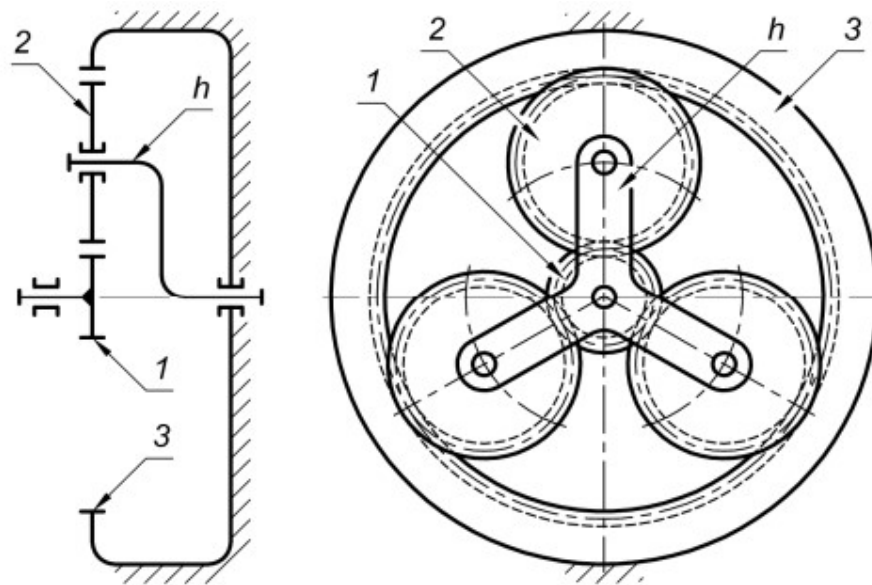


Figure 5.4. Planetary gearbox scheme.

The sun gear 1, planet gear 2, ring gear 3 (which is fixed, meaning $3 = 0$), and planet carrier h are the parts of the chosen planetary gearbox. The sun gear provides energy, which the planet carrier absorbs. For transmission ratios between 3 and 9, this kind of planetary gearbox has a high coefficient of efficiency. It is easy to make, has a noticeably low bulk, and compact size. In our case, we have used a planetary gear DC motor which has a gear ratio of 26.9: 1. If the information about transmission ratio, tooth helix angle are known to us, then by these information we can calculate the number of teeth, calculate number of planet gears, and also calculate the gear modules and its diameters. Which can be used for design purpose of the gears and can be used for minimal backlash.

5.4 Sensor Nonlinearity

In wheel mobile robots, sensor nonlinearity refers to deviations in sensor measurements from ideal linear response. The control and navigation of the robot may be severely impacted by this. Wheel encoders, which are frequently used for odometry, may give nonlinear readings as a result of mechanical flaws or restrictions in the signal processing. This could result in inaccurate calculations of the robot's position and speed.

- **Motion Sensor Nonlinearity:** Inertial sensors, such as gyroscopes and accelerometers, may display nonlinear behavior, particularly at high angular rates or accelerations. The precision of motion control and orientation estimates may be impacted by nonlinearity in these sensors.
- **Range Sensors:** Sensors like ultrasonic or LIDAR range finders can have nonlinear responses, particularly at varying distances from objects. This can affect obstacle detection and mapping accuracy.
- **Compass sensors:** Magnetic interference can cause nonlinearity in magnetometer sensors used to measure compass headings, which can result in erroneous direction determination.

To mitigate sensor nonlinearity, various calibration techniques and sensor fusion methods are employed. Calibration involves characterizing and compensating for sensor-specific nonlinearities. Sensor fusion combines data from multiple sensors to enhance accuracy and reduce the impact of individual sensor nonlinearities. Additionally, advanced control algorithms, such as nonlinear control approaches, can help account for sensor nonlinearity when making control decisions. Managing sensor nonlinearity is crucial for ensuring the reliable and precise operation of wheel mobile robots in various applications, including autonomous navigation and robotics research.

5.5 Sensor Nonlinearity

The interaction between the wheels and the ground often exhibits nonlinear behaviour, posing challenges in the precise prediction of the resultant forces governing wheel motion. Friction nonlinearity in wheeled mobile robots is a complex and crucial aspect of their operation, requiring dynamic modelling, control strategies, and force distribution estimation to ensure effective and stable movement on various surfaces

Chapter 6: Effect of Nonlinearities, Measurement Uncertainties and Noise in Kinematic based Odometry Physical Nonlinearities in WMR

Nonlinearities, measurement uncertainties, and noise can significantly impact the accuracy of kinematic-based odometry systems. The results are as follows:

6.1 Nonlinearities

In the previous chapter we've discussed about the different types of Nonlinearities acting on the wheeled mobile robot and its effect on the robot's Kinematic also discussed there.

Kinematic models, which assume linear relationships between input commands and consequent motion, may simplify a vehicle's movements. In practice, these relationships may not be linear, particularly during sudden movements or at high speeds. Over time, this may result in increased cumulative errors and decreased odometry accuracy. Even the Kinematic model of the robot is nonlinear.

A two-wheeled mobile robot's kinematic model is intrinsically nonlinear. This is due to the fact that the mobility of the robot is dependent on elements such as wheel speeds and turning angles, which are together via trigonometric functions. Because of this intricacy, precise position and orientation control is difficult, especially at different speeds.

6.1 Uncertainty in WMR

Uncertainty arises if the robot lacks critical information for carrying out its task. It arises from five different factors:

- **Environments:** Physical worlds are inherently unpredictable. While the degree of uncertainty in well-structured environments such as assembly lines is small, environments such as highways and private homes are highly dynamic and unpredictable.
- **Sensors:** Sensors are inherently limited in what they can perceive. Limitations arise from two primary factors. First, range and resolution of a sensor is subject to physical laws. For example, Cameras can't see through walls, and even within the perceptual range the spatial

resolution of camera images is limited. Second, sensors are subject to noise, which perturbs sensor measurements in unpredictable ways and hence limits the information that can be extracted from sensor measurements.

- **Robots:** Robot actuation involves motors that are, at least to some extent, unpredictable, due effects like control noise and wear-and-tear. Low-cost robots like mobile robots, can be extremely inaccurate.
- **Models:** Models are inherently inaccurate. Models are abstractions of the real world. As such, they only partially model the underlying physical processes of the robot and its environment. Model errors are a source of uncertainty that has largely been ignored in robotics, despite the fact that most robotic models used in state-of-the-art robotics systems are rather crude.
- **Computation:** Robots are real-time systems, which limit the amount of computation that can be carried out. Many state-of-the-art algorithms (such as most of the algorithms described in this book) are approximate, achieving timely response through sacrificing accuracy. All of these factors give rise to uncertainty. Traditionally, such uncertainty has mostly been ignored in robotics. However, as robots are moving away from factory floors into increasingly unstructured environments, the ability to cope with uncertainty is critical for building successful robots.

Uncertainty in wheeled mobile robot's systems from several sources, including unknown parameters, modelling inaccuracies, and nonlinearities. These uncertainties can significantly impact a robot's performance and control. Researchers have devised various strategies to tackle this issue, with one prominent approach being the implementation of adaptive control schemes. These schemes aim to diminish the influence of unknown dynamic parameters and uncertainties, enabling robots to effectively trace desired trajectories.

Another strategy is to create nonlinear robust control laws that, even in the presence of modelling uncertainties, can quickly converge tracking errors between the controlled robot and the intended trajectory. A methodical approach has also been used to create a modelling structure for wheeled mobile robots that can deal with noise, disturbances, and dynamic model uncertainty.

6.1 Effect of Noise in WMR

The term "noise" in the context of wheeled mobile robots refers to unwelcome or unplanned disturbances and errors that may affect the operation and control of the robots. Different aspects of robot performance can be impacted by this noise, which can come from a variety of sources:

6.1.1 Sensor Noise

Sensor noise is a critical aspect of sensor technology and measurement systems. It refers to the unwanted and random variations in the output signal of a sensor that are not related to the quantity being measured. This noise can arise from various sources and has several important implications:

- **Origins of Sensor Noise:** There are several things that might cause sensor noise, such as electrical parts, the environment, manufacturing flaws, and the intrinsic limitations of the sensor technology itself.
- **Types of Sensor Noise:** Different types of sensor noise, including thermal noise, flicker noise, and shot noise, can be present. Each type has unique traits that affect how well sensors work.
- **Effects on Measurement:** Sensor noise can cause measurements to be inaccurate and imprecise. It may lower the accuracy and dependability of sensor readings, lowering the caliber of sensor-derived data.
- **Noise reduction:** In scientific and technical applications, sensor noise reduction attempts are conducted. To reduce the impacts of noise, methods include signal processing, filtering, and the use of specific sensor types.

For precise and dependable sensor data, wheeled mobile robots must eliminate sensor noise. Here are a few tactics:

- **Sensor Calibration:** To correct any systematic flaws and guarantee measurement accuracy, calibrate sensors on a regular basis.
- **Signal filtering:** To eliminate high-frequency noise in sensor data, use signal processing techniques like low-pass filters.
- **Data Fusion:** Data fusion is the process of combining data from various sensors to increase accuracy and lessen the impacts of noise.

- **Environmental Control:** Reduce environmental noise-causing elements like electromagnetic interference and vibrations.
- **Sensor Selection:** Choose sensors with lower noise characteristics for specific applications when possible.

6.1.2 Acoustic Noise

Acoustic noise is produced by mobile robots that have motors and actuators. Managing and minimizing this noise can be essential in scenarios where stealth or reduced disturbance is required, such as surveillance or quiet environments.

Wheeled mobile robots can face major difficulties when it comes to acoustic noise, particularly in situations where stealth or low noise levels are necessary. The following are some methods and tools for reducing acoustic noise:

- **Acoustic Signature Reduction:** Research and technologies exist for reducing the acoustic signature of mobile robots. This entails reducing mechanical vibrations and creating robots with noisier propulsion systems.
- **Incorporation of Acoustic Sensors:** Some robots incorporate acoustic sensors to detect and analyse environmental noise. Robots can modify their behaviour with the aid of these sensors to reduce noise production. Robots that can detect and react to their own noise emissions will be more socially acceptable in human surroundings, according to research on motion-induced acoustic noise awareness. Robots can choose paths that minimize acoustic noise, especially in outdoor or rugged terrain, with the use of terrain classification frameworks that use sensor data.
- **Acoustic Proximity Sensors:** Acoustic proximity sensors are appropriate for noise-sensitive applications because they may be utilized to detect close objects or obstacles without producing electromagnetic interference.

6.1.3 Communication and Control Noise

The robot's ability to perform precise motions and reliably receive commands can be impacted by noise in control signals or communication lines. Control responses may become unstable or take longer to respond.

6.1.4 Processing and Latency Noise

The robot's real-time decision-making, which is essential for tasks like autonomous navigation and interaction with the environment, can be impacted by delays and noise in the processing of data or control algorithms.

Accurate control and navigation of wheeled mobile robots depend on the elimination of processing and latencies noise. Following are some tactics based on the sources mentioned:

- To reduce latencies and noise in vision-based control systems, consider hardware and image processing algorithm optimization.
- Use fuzzy control techniques to improve tracking accuracy and reduce noise for mobile robots that depend on sound source tracking, such as those used in monitoring or surveillance.
- Conduct simulation and experimental investigations to pinpoint and resolve the latency and noise problems in control systems.

These strategies can help mitigate the effects of processing and latencies noise, improving the overall performance of wheeled mobile robots. Additionally, to continuously adjust the robot's behaviour and responses in dynamic situations, real-time monitoring and adaptive control techniques would be required.

Chapter 7: Conclusion

In the presented work, we achieved Indoor Localization using the Kinematic model of Differential Drive Wheeled Mobile Robot. The Kinematic model of Differential Drive Wheeled Mobile Robot is a fundamental tool in robotics and engineering, playing a crucial role in robot controls, path planning, simulation and various other applications related to the motion and navigation in wheeled mobile robots.

We found the radius of the wheel using screw gauge and also found the distance between two wheels by using a linear scale. Then by using gyroscope of IMU we found the orientation of the robot. Then feed this data, we got the kinematic equations of the Differential Drive Wheeled Mobile Robot which is already discussed on the Chapter Number 3 in our thesis.

However, the Kinematic model can be used as a component in the indoor localization of the wheeled mobile robots. The Kinematic model predicts how the robots pose changes with the help of wheel movements and wheel velocities. Then using Dead Reckoning method, we estimate the robots pose using its Wheel encoder.

To improve the accuracy of the Indoor Localization, the Kinematic model is often combined with the sensor data, mainly from the other sources. Here we used Hall Effect Magnetic Encoder OE37 and Adafruit BNO005 Inertial Measurement Unit (IMU) for accuracy of the localization.

Encoder measure the rotation of the robot's wheel and IMU provides the information about the robots acceleration and angular velocities. By integrate the information over time we get the position and orientation of the Wheeled Mobile Robot. Using these data, the position and the orientation of the wheeled mobile robot have been obtained with the help of Extended Kalman Filter.

From observation, we can say that robot does not follow the given command. There is a slight deviation of its trajectory from the given ones. Then to mitigate this error, we try to working on the Robot Model.

We found that the diameter of the robot's wheel are not same, and the DC motor which is used as an actuator are not same specifications. We're trying to measure the no-load resistance and current of the DC motor by using a multi meter, however the readings are not same for the two motors.

Even, we found that sensor measurement and wheel encoder accumulate errors due to the various factors like Wheel Slippage, Sensor Noise and mainly for the Backlash of the Motor.

From these above mentioned studies we can conclude that several nonlinearities are acted in this Differential Drive Wheeled Mobile Robot's Model. To mitigate these we have proposed feedback control, Nonlinear Observer Design and a Robust Controlling Algorithm in the Future Scope of this thesis.

Chapter 8: Future Scope

The study of nonlinearities in wheeled mobile robots and their impact on kinematic offers a vast and promising future scope that can drive innovation and advancements in the field of robotics. Some key areas of future exploration include:

1. Advanced Sensor Technologies: The development of more sophisticated and robust sensors, such as high-resolution wheel encoders, proprioceptive sensors, and terrain sensors, can significantly enhance our ability to detect and compensate for nonlinearities. These sensors could provide real-time feedback to improve motion control and localization accuracy.

2. Feedback Loop Design: We can design a feedback loop for controlling the wheel velocities of the Differential Drive Wheeled Mobile Robot, by which we can get a better error free localization in indoor arena and get a better performance.

3. Nonlinear Observer Design: Nonlinear observer design is a crucial aspect of control theory and system engineering. It involves creating a mathematical model or algorithm that estimates the internal state variables of a dynamic system, even when the system itself is nonlinear and only partially observable through measurements. Observers are essential for various applications, including state estimation in robotics, aerospace, automotive systems, and industrial processes.

Designing a nonlinear observer can be a complex and iterative process, but it is essential for accurately estimating the state of nonlinear systems. The choice of observer type and its parameters should be carefully tailored to the specific characteristics of the system and the requirements of the application.

4. Wi-Fi Positioning System: We can use Wi-Fi for the indoor localization of the robot and by comparing the data extracted from Wi-Fi module, with the encoder and IMU, We will mitigate the nonlinearities from the system and get a better localization of the robot.

5. Robust Algorithm Design: Robust algorithm for Wheeled Mobile Robots are crucial to ensure that the robot can operate effectively in diverse and challenging environments while handling uncertainties and nonlinearities in the Wheeled Mobile Robot.

We can go for a better data fusion techniques and for the algorithms that optimized a better energy estimations, and algorithms like Visual-Inertial odometry can provide a reliable position estimation.

In summary, the future scope of research on nonlinearities in wheeled mobile robots and their effects on kinematic is expansive and dynamic. It offers numerous opportunities for collaboration, technological innovation, and the development of more capable and versatile robots that can operate effectively in a wide range of environments and tasks.

References

- [1] G. Campion and W. Chung, “Wheeled Rob 17. Wheeled Robots,” 2008.
- [2] B. Diriba, H., P. Wang, and Z., “Design and Control for Differential Drive Mobile Robot,” Tiranjin, Oct. 2017. [Online]. Available: www.ijert.org.
- [3] G. Dudek and M. Jenkin, “Computational Principles of Mobile Robotics,” New York, Apr. 2010.
- [4] J.Huang, S.Junginger, H.Liu, and K.Thurow, “Indoor Positioning Systems of Mobile Robots: A Review,” *Robotics*, pp. 1-28, vol. 12 ,no.2, 2023.
- [5] J. Qian, B. Zi, D. Wang, Y. Ma, and D. Zhang, “The design and development of an Omni-Directional mobile robot oriented to an intelligent manufacturing system,” *Sensors (Switzerland)*, vol. 17, no. 9, Sep. 2017, doi: 10.3390/s17092073.
- [6] S. Khatoon, M. Istiyaque, A. S. Wani, and M. Shahid, “Design Kinematic and Control for a Differential Drive Mobile Robot,” Singapore: Springer, Singapore, Apr. 2021, pp. 189–196.
- [7] “Wheeled Mobile Robot: Topics by WorldWideScience.org,” <https://worldwidescience.org/topicpages/w/wheeled+mobile+robot.html>.
- [8] L. Sun, D. Adolfsson, M. Magnusson, H. Andreasson, I. Posner, and T. Duckett, Localising Faster: Efficient and precise lidar-based robot localisation in large-scale environments. 2020. doi: 10.0/Linux-x86_64.
- [9] H. Zhao, C. Luo, Y. Xu, and J. Li, “Differential Steering Control for 6×6 Wheel-drive Mobile Robot,” in 2021 26th International Conference on Automation and Computing (ICAC), 2021, pp. 1–6. doi: 10.23919/ICAC50006.2021.9594210.
- [10] B. S. Malu, J. Majumdar, S. Malu α , and J. Majumdar σ , “Kinematic, Localization and Control of Differential Drive Mobile Robot Global Journal of Researches in Engineering: H Kinematic, Localization and Control of Differential Drive Mobile Robot,” Type: Double Blind Peer Reviewed International Research Journal Publisher: Global Journals Inc, vol. 14, 2014.

- [11] S. F. R. Alves, J. M. Rosario, H. F. Filho, L. K. A. Rincon, and R. A. T. Yamasaki, "Conceptual Bases of Robot Navigation Modeling, Control and Applications," in *Advances in Robot Navigation*, A. Barrera, Ed., Rijeka: IntechOpen, 2011. doi: 10.5772/20955.
- [12] K. Kothandaraman, "Motion Planning and Control of Differential Drive Robot," 2016. [Online].
- [13] K. Kothandaraman, "Motion Planning and Control of Differential Drive Robot," 2016.
- [14] J. Lin, J. Peng, Z. Hu, X. Xie, and R. Peng, "ORB-SLAM, IMU and Wheel Odometry Fusion for Indoor Mobile Robot Localization and Navigation," *Academic Journal of Computing & Information Science*, vol. 3, pp. 131–141, doi: 10.25236/AJCIS.2020.030114.
- [15] A. V. Chavan and J. L. Minase, "DESIGN OF A DIFFERENTIAL DRIVE MOBILE ROBOT PLATFORM FOR USE IN CONSTRAINED ENVIRONMENTS," *INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]*, vol. 2, no. 6, Jun. 2015.
- [16] E. Alhamdi and R. Hedjar, "Mobile Robot Localization Using Extended Kalman Filter," *Riyadh: 3rd ICCAIS 2020 : International Conference on Computer Applications & Information Security*, Mar. 2020.
- [17] Y. Liu, Y. Ou, and W. C. Han, "Mobile Robot Localization Based on Optical Sensor," Yang Liu, YongSheng Ou*, and WeiChao Han, Eds., Russia: *IEEE International Conference on Real-time Computing and Robotics*, Sep. 2019.
- [18] X. Li, D. Wang, Y. Liu, and Z. Peng, "Mobile Robot Localization Using Soft Sensor," in *Proceedings - 5th International Conference on Automation, Control and Robotics Engineering, CACRE 2020*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 66–70. doi: 10.1109/CACRE50138.2020.9229997.
- [19] O. S. Shipitko, M. P. Abramov, A. S. Lukoyanov, E. I. Panfilova, I. A. Kunina, and A. S. Grigoryev, "Edge detection based mobile robot indoor localization," in *Eleventh International Conference on Machine Vision (ICMV 2018)*, A. Verikas, D. P. Nikolaev, P. Radeva, and J. Zhou, Eds., SPIE, 2019, p. 110412V. doi: 10.1117/12.2522788.

- [20] X. Llado, J. Aulinas, X. Lladó, J. Salvi, and Y. R. Petillot, "SLAM base Selective Submap Joining for the Victoria Park Dataset Tissue segmentation and quantification on brain MRI using convolutional neural networks View project USMART View project SLAM based Selective Submap Joining for the Victoria Park Dataset ☆," 2010. [Online]. Available: <https://www.researchgate.net/publication/228696361>
- [21] E. O. Meshkovskiy, V. Y. Frolov, and A. D. Kurmashev, "Nonlinear control of electric drive system of four-wheel mobile robot with two differential drive units," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Feb. 2021. doi: 10.1088/1742-6596/1753/1/012031.
- [22] I. F. Okuyama, M. R. O. A. Maximo, and R. J. M. Afonso, "Minimum-Time Trajectory Planning for a Differential Drive Mobile Robot Considering Non-slipping Constraints," *Journal of Control, Automation and Electrical Systems*, vol. 32, pp. 120–131, 2020, [Online]. Available: <https://api.semanticscholar.org/CorpusID:228841797>
- [23] T. G. Alves, W. F. Lages, and R. V. B. Henriques, "Non-linear Pose Stabilization Controller for a Differential-Drive Mobile Robot: Optimization-Based Controller Tuning," *IFAC-PapersOnLine*, vol. 51, no. 22, pp. 536–541, 2018, doi: <https://doi.org/10.1016/j.ifacol.2018.11.576>.
- [24] M. Crenganis, C. Biris, and C. Girjob, "Mechatronic Design of a Four-Wheel drive mobile robot and differential steering," *MATEC Web of Conferences*, vol. 343, p. 08003, 2021, doi: 10.1051/mateconf/202134308003.
- [25] R. Beniak and T. Pyka, "Mobile robot with non-slip castor wheel," in *2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, 2017, pp. 202–205. doi: 10.1109/MMAR.2017.8046824.
- [26] I. Anvari, A. A. Rodriguez, J. Si, and K. Tsakalis, "Non-holonomic Differential Drive Mobile Robot Control & Design : Critical Dynamics and Coupling Constraints by the Graduate Supervisory Committee," 2013.
- [27] D. Wang and C. B. Low, "Modeling and analysis of skidding and slipping in wheeled mobile robots: Control design perspective," *IEEE Trans. on Robotics*, vol. 24(3), pp. 676–687, 2008.

- [28] R. Balakrishna and A. Ghosal, "Modeling of slip for wheeled mobile robots," IEEE Trans. on Robotics and Automation, vol. 11(1), pp. 126-132, 1995.
- [29] C. B. Low and D. Wang, "GPS-based path following control for a car-like wheeled mobile robot with skidding and slipping," IEEE Trans. on Control Systems Technology, vol. 16(2), pp. 340-347, 2008.
- [30] O. A. Ani, H. Xu and G. Zhao, "Analysis and modelling of slip for a five-wheeled mobile robot (WMR) in an uneven terrain," IEEE Int. Conf. on Mechatronics and Automation (ICMA), Beijing, China, 2011, pp. 154-159.
- [31] E. F. Kececi and G. Tao, "Adaptive vehicle skid control," Elsevier J. of Mechatronics, vol. 16(5), pp. 291-301, 2006.
- [32] M. Amodeo, A. Ferrara, R. Terzaghi, and C. Vecchio, "Wheel slip control via second-order sliding-mode generation," IEEE Trans. on Intelligent Transportation Systems, vol. 11(1), pp. 122-131, 2010.
- [33] S. J. Yoo, "Adaptive tracking control for a class of wheeled mobile robots with unknown skidding and slipping," IEEE J. of Control Theory & Applications, vol. 4(10), pp. 2109-2119, 2010.
- [34] W. Dong, "Control of uncertain wheeled mobile robots with slipping," 49th IEEE Int. Conf. on Decision and Control (CDC), Georgia, USA, 2010, pp. 7190-7195.
- [35] G. A. Magallan, C. H. De Angelo and G. O. Garcia, "Maximization of the traction forces in a 2WD electric vehicle," IEEE Trans. on Vehicular Technology, vol. 60(2), pp. 369-380, 2011.
- [36] Y. Tian and N. Sarkar, "Formation control of mobile robots subject to wheel slip," IEEE Int. Conf. on Robotics and Automation (ICRA), St. Paul, USA 2012, pp. 4553-4558.
- [37] V. Ćirović, and D. Aleksendrić, "Adaptive neuro-fuzzy wheel slip control," Elsevier J. of Expert Systems with Applications, March 2013 (In press).

- [38] V. Ćirović, D. Aleksendrić and D. Smiljanić, “Longitudinal wheel slip control using dynamic neural networks,” Elsevier J. of Mechatronics, vol. 23(1), pp. 135-146, 2013.
- [39] Robu,” Orange OE-37 Encoder Two Channel Encoder”.
- [40] Robu,” Orange HD Planetary Gear DC Motor”.
- [41] Bosch,” BNO055 Intelligent 9-axis absolute orientation sensor”, BST-BNO055-DS000-12, November 2014.

Appendix A : Arduino Code

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

```
/* This driver uses the Adafruit unified sensor library
(Adafruit_Sensor),
which provides a common 'type' for sensor data and some helper
functions.
```

To use this driver, you will also need to download the Adafruit_Sensor library and include it in your libraries folder.

You should also assign a unique ID to this sensor for use with the Adafruit Sensor API so that you can identify this particular sensor in any data logs, etc. To assign a unique ID, simply provide an appropriate value in the constructor below (12345 is used by default in this example).

Connections
=====

Connect SCL to analog 5
Connect SDA to analog 4
Connect VDD to 3.3-5V DC
Connect GROUND to common ground

History
=====

2015/MAR/03 - First release (KTOWN)
*/

```
/* Set the delay between fresh samples */
uint16_t BNO055_SAMPLERATE_DELAY_MS = 100;
```

```
// Check I2C device address and correct line below (by default
address is 0x29 or 0x28)
// id, address
Adafruit_BNO055 bno = Adafruit_BNO055(55, 0x28, &Wire);
```

```
/*
This program can control the robot remotly.
An android app is beenbuild to control the robot called bot-
controller.
*/
#define Encoder_L_output_A 18 // pin2 of the Arduino
```

```

#define Encoder_L_output_B 17 // pin 3 of the Arduino
#define Encoder_R_output_A 2 // pin2 of the Arduino
#define Encoder_R_output_B 3 // pin 3 of the Arduino

// these two pins has the hardware interrupts as well.

int Count_L_pulses = 0;
int Count_R_pulses = 0;

int speedL=255;//67;
int speedR=255;//88;
//speed=60;

int motor1Pin1 = 7; // pin 2 on L293D IC
int motor1Pin2 = 8; // pin 7 on L293D IC
int enable1Pin = 9; // pin 1 on L293D IC
int motor2Pin1 = 10; // pin 10 on L293D IC
int motor2Pin2 = 11; // pin 15 on L293D IC
int enable2Pin = 12; // pin 9 on L293D IC
int state;

int flag=0; //makes sure that the serial only prints once the state
int stateStop=0;

void setup() {

Serial.begin(9600);

while (!Serial) delay(10); // wait for serial port to open!

Serial.println("Orientation Sensor Test"); Serial.println("");

/* Initialise the sensor */
if (!bno.begin())
{
  /* There was a problem detecting the BNO055 ... check your
  connections */
  Serial.print("Oops, no BNO055 detected ... Check your wiring or
  I2C ADDR!");
  while (1);
}

delay(1000);

pinMode(Encoder_L_output_A, INPUT); // sets the Encoder_output_A pin
as the input

```

```

pinMode(Encoder_L_output_B, INPUT); // sets the Encoder_output_B pin
as the input
pinMode(Encoder_R_output_A, INPUT); // sets the Encoder_output_A pin
as the input
pinMode(Encoder_R_output_B, INPUT); // sets the Encoder_output_B pin
as the input

attachInterrupt(digitalPinToInterrupt(Encoder_L_output_A), DC_L_Motor
_Encoder, RISING);
attachInterrupt(digitalPinToInterrupt(Encoder_R_output_A), DC_R_Motor
_Encoder, RISING);

// sets the pins as outputs:

pinMode(motor1Pin1, OUTPUT);
pinMode(motor1Pin2, OUTPUT);
pinMode(enable1Pin, OUTPUT);
pinMode(motor2Pin1, OUTPUT);
pinMode(motor2Pin2, OUTPUT);
pinMode(enable2Pin, OUTPUT);
// sets enable1Pin and enable2Pin high so that motor can turn on:
digitalWrite(enable1Pin, HIGH);
digitalWrite(enable2Pin, HIGH);
// initialize serial communication at 9600 bits per second:

}

void loop() {

Serial.flush();
//could add VECTOR_ACCELEROMETER,
VECTOR_MAGNETOMETER, VECTOR_GRAVITY...
sensors_event_t orientationData , angVelocityData , linearAccelData,
magnetometerData, accelerometerData, gravityData;
bno.getEvent(&orientationData, Adafruit_BNO055::VECTOR_EULER);
bno.getEvent(&angVelocityData, Adafruit_BNO055::VECTOR_GYROSCOPE);
bno.getEvent(&linearAccelData, Adafruit_BNO055::VECTOR_LINEARACCEL);
bno.getEvent(&magnetometerData,
Adafruit_BNO055::VECTOR_MAGNETOMETER);
bno.getEvent(&accelerometerData,
Adafruit_BNO055::VECTOR_ACCELEROMETER);
bno.getEvent(&gravityData, Adafruit_BNO055::VECTOR_GRAVITY);

printEvent(&orientationData);
printEvent(&angVelocityData);
printEvent(&linearAccelData);
printEvent(&magnetometerData);
printEvent(&accelerometerData);
printEvent(&gravityData);

```



```

int8_t boardTemp = bno.getTemp();
Serial.println();
Serial.print(F("temperature: "));
Serial.println(boardTemp);

uint8_t system, gyro, accel, mag = 0;
bno.getCalibration(&system, &gyro, &accel, &mag);
Serial.println();
Serial.print("Calibration: Sys=");
Serial.print(system);
Serial.print(" Gyro=");
Serial.print(gyro);
Serial.print(" Accel=");
Serial.print(accel);
Serial.print(" Mag=");
Serial.println(mag);

Serial.println("--");
delay(BNO055_SAMPLERATE_DELAY_MS);
//Motor encoder data
Serial.println("Result_L: ");
Serial.println(Count_L_pulses);
Serial.println("Result_R: ");
Serial.println(Count_R_pulses);
//Motor encoder data

//if some data is sent, reads it and saves in state
if(Serial.available() > 0){
state = Serial.read();
flag=0;
}
// if the state is '5' the DC motor will go forward
if (state == '5') {

analogWrite(enable1Pin, speedL);          //sets the motors speed
analogWrite(enable2Pin, speedR);          //sets the motors speed

digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, HIGH);

if(flag == 0){

```

```

Serial.println("Go Forward!");
flag=1;
}

}

// if the state is '4' the motor will turn left
else if (state == '4') {
analogWrite(enable1Pin, speedL);      //sets the motors speed
analogWrite(enable2Pin, speedR);      //sets the motors speed

digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, HIGH);
digitalWrite(motor2Pin2, LOW);

if(flag == 0){
Serial.println("Turn LEFT");
flag=1;
}
delay(1100);
state=3;
stateStop=1;
}
// if the state is '3' the motor will Stop
else if (state == '3' || stateStop == 1) {
analogWrite(enable1Pin, speedL);      //sets the motors speed
analogWrite(enable2Pin, speedR);      //sets the motors speed

digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
if(flag == 0){
Serial.println("STOP!");
flag=1;
}
stateStop=0;
}
// if the state is '2' the motor will turn right

else if (state == '2') {
analogWrite(enable1Pin, speedL);      //sets the motors speed
analogWrite(enable2Pin, speedR);      //sets the motors speed

digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, HIGH);

```

```

digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, HIGH);
if(flag == 0){
Serial.println("Turn RIGHT");
flag=1;
}
delay(1090);
state=3;
stateStop=1;
}
// if the state is '1' the motor will Reverse
else if (state == '1') {
analogWrite(enable1Pin, speedL);          //sets the motors speed
analogWrite(enable2Pin, speedR);          //sets the motors speed

digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, HIGH);
digitalWrite(motor2Pin1, HIGH);
digitalWrite(motor2Pin2, LOW);
if(flag == 0){
Serial.println("Reverse!");
flag=1;
}
}
//For debugging purpose
//Serial.println(state);
}

/*
 * The purpose of this code is to count the output pulses or
 * the encoder outputs as you rotate the Motor shaft. You can run
the
 * same code on the Arduino Uno, Arduino Nano, Arduino Mega, etc.
 */

void printEvent(sensors_event_t* event)
{
    double x = -1000000, y = -1000000 , z = -1000000; //dumb values,
easy to spot problem
    if (event->type == SENSOR_TYPE_ACCELEROMETER)
    {
        Serial.print("Accl:");
        x = event->acceleration.x;
        y = event->acceleration.y;
        z = event->acceleration.z;
    }
}

```

```

else if (event->type == SENSOR_TYPE_ORIENTATION)
{
    Serial.print("Orient:");
    x = event->orientation.x;
    y = event->orientation.y;
    z = event->orientation.z;
}
else if (event->type == SENSOR_TYPE_MAGNETIC_FIELD)
{
    Serial.print("Mag:");
    x = event->magnetic.x;
    y = event->magnetic.y;
    z = event->magnetic.z;
}
else if (event->type == SENSOR_TYPE_GYROSCOPE)
{
    Serial.print("Gyro:");
    x = event->gyro.x;
    y = event->gyro.y;
    z = event->gyro.z;
}
else if (event->type == SENSOR_TYPE_ROTATION_VECTOR)
{
    Serial.print("Rot:");
    x = event->gyro.x;
    y = event->gyro.y;
    z = event->gyro.z;
}
else if (event->type == SENSOR_TYPE_LINEAR_ACCELERATION)
{
    Serial.print("Linear:");
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
}
else if (event->type == SENSOR_TYPE_GRAVITY)
{
    Serial.print("Gravity:");
    x = event->acceleration.x;
    y = event->acceleration.y;
    z = event->acceleration.z;
}
else
{
    Serial.print("Unk:");
}

Serial.print("\tx= ");

```

```

    Serial.print(x);
    Serial.print(" |\ty= ");
    Serial.print(y);
    Serial.print(" |\tz= ");
    Serial.println(z);
}

void DC_L_Motor_Encoder() {
    int b = digitalRead(Encoder_L_output_B);
    if(b > 0) {
        Count_L_pulses++;
    }
    else{
        Count_L_pulses--;
    }
}

void DC_R_Motor_Encoder() {
    int b = digitalRead(Encoder_R_output_B);
    if(b > 0) {
        Count_R_pulses++;
    }
    else{
        Count_R_pulses--;
    }
}

```

Appendix B : Python Code

```
import numpy as np
import matplotlib.pyplot as plt

# Description: Extended Kalman Filter example (two-wheeled mobile
robot)

# Supress scientific notation when printing NumPy arrays
np.set_printoptions(precision=3,suppress=True)

# A matrix
# 3x3 matrix -> number of states x number of states matrix
# Expresses how the state of the system [x,y,yaw] changes
# from k-1 to k when no control command is executed.
# Typically, a robot on wheels only drives when the wheels are told
to turn.
# For this case, A is the identity matrix.
# A is sometimes F in the literature.
A_k_minus_1 = np.array([[1.0, 0, 0],
                        [0, 1.0, 0],
                        [0, 0, 1.0]])

# Noise applied to the forward kinematic (calculation
# of the estimated state at time k from the state
# transition model of the mobile robot). This is a vector
# with the number of elements equal to the number of states
process_noise_v_k_minus_1 = np.array([0.035,0.035,0.0032])

# State model noise covariance matrix Q_k
# When Q is large, the Kalman Filter tracks large changes in
# the sensor measurements more closely than for smaller Q.
# Q is a square matrix that has the same number of rows as states.
Q_k = np.array([[1.0, 0, 0],
                [0, 1.0, 0],
                [0, 0, 1.0]])

# Measurement matrix H_k
# Used to convert the predicted state estimate at time k
# into predicted sensor measurements at time k.
# In this case, H will be the identity matrix since the
# estimated state maps directly to state measurements from the
# odometry data [x, y, yaw]
# H has the same number of rows as sensor measurements
# and same number of columns as states.
H_k = np.array([[1.0, 0, 0],
                [0, 1.0, 0],
```

```

        [ 0, 0, 1.0]])

# Sensor measurement noise covariance matrix R_k
# Has the same number of rows and columns as sensor measurements.
# If we are sure about the measurements, R will be near zero.
R_k = np.array([[1.0, 0, 0],
                [ 0, 1.0, 0],
                [ 0, 0, 1.0]])

# Sensor noise. This is a vector with the
# number of elements equal to the number of sensor measurements.
sensor_noise_w_k = np.array([0.08,0.03,0.04])

def getB(yaw, deltak):
    """
    Calculates and returns the B matrix
    3x2 matrix -> number of states x number of control inputs
    The control inputs are the forward speed and the
    rotation rate around the z axis from the x-axis in the
    counterclockwise direction.
    [v,yaw_rate]
    Expresses how the state of the system [x,y,yaw] changes
    from k-1 to k due to the control commands (i.e., control input).
    :param yaw: The yaw angle (rotation angle around the z axis) in
rad
    :param deltak: The change in time from time step k-1 to k in sec
    """
    B = np.array([ [np.cos(yaw)*deltak, 0],
                   [np.sin(yaw)*deltak, 0],
                   [0, deltak]])

    return B

def ekf(z_k_observation_vector, state_estimate_k_minus_1,
        control_vector_k_minus_1, P_k_minus_1, dk):
    """
    Extended Kalman Filter. Fuses noisy sensor measurement to
    create an optimal estimate of the state of the robotic system.

    INPUT
    :param z_k_observation_vector The observation from the
Odometry
        3x1 NumPy Array [x,y,yaw] in the global reference frame
        in [meters,meters,radians].
    :param state_estimate_k_minus_1 The state estimate at time
k-1
        3x1 NumPy Array [x,y,yaw] in the global reference frame
        in [meters,meters,radians].

```

```

        :param control_vector_k_minus_1 The control vector applied
at time k-1
        3x1 NumPy Array [v,v,yaw rate] in the global reference
frame
        in [meters per second,meters per second,radians per
second].
        :param P_k_minus_1 The state covariance matrix estimate at
time k-1
        3x3 NumPy Array
        :param dk Time interval in seconds

OUTPUT
        :return state_estimate_k near-optimal state estimate at time
k
        3x1 NumPy Array ---> [meters,meters,radians]
        :return P_k state covariance_estimate for time k
        3x3 NumPy Array
"""
##### Predict #####
# Predict the state estimate at time k based on the state
# estimate at time k-1 and the control input applied at time k-
1.
state_estimate_k = A_k_minus_1 @ (
    state_estimate_k_minus_1) + (
    getB(state_estimate_k_minus_1[2],dk)) @ (
    control_vector_k_minus_1) + (
    process_noise_v_k_minus_1)

print(f'State Estimate Before EKF={state_estimate_k}')

plt.plot(state_estimate_k[0],state_estimate_k[1],color='red',marker=
'o')
plt.pause=0.5
# Predict the state covariance estimate based on the previous
# covariance and some noise
P_k = A_k_minus_1 @ P_k_minus_1 @ A_k_minus_1.T + (
    Q_k)

##### Update (Correct) #####
# Calculate the difference between the actual sensor
measurements
# at time k minus what the measurement model predicted
# the sensor measurements would be for the current timestep k.
measurement_residual_y_k = z_k_observation_vector - (
    (H_k @ state_estimate_k) + (
    sensor_noise_w_k))

print(f'Observation={z_k_observation_vector}')

```



```

plt.plot(z_k_observation_vector[0],z_k_observation_vector[1],color='
blue',marker='*')
    plt.pause(0.5)
    # Calculate the measurement residual covariance
    S_k = H_k @ P_k @ H_k.T + R_k

    # Calculate the near-optimal Kalman gain
    # We use pseudoinverse since some of the matrices might be
    # non-square or singular.
    K_k = P_k @ H_k.T @ np.linalg.pinv(S_k)

    # Calculate an updated state estimate for time k
    state_estimate_k = state_estimate_k + (K_k @
measurement_residual_y_k)

    # Update the state covariance estimate for time k
    P_k = P_k - (K_k @ H_k @ P_k)

    # Print the best (near-optimal) estimate of the current state of
the robot
    print(f'State Estimate After EKF={state_estimate_k}')

plt.plot(state_estimate_k[0],state_estimate_k[1],color='green',marke
r='*')
    plt.pause(0.5)
    # Return the updated state and covariance estimates
    return state_estimate_k, P_k

def main():

    # We start at time k=1
    k = 1

    # Time interval in seconds
    dk = 1

    # Create a list of sensor observations at successive timesteps
    # Each list within z_k is an observation vector.

    z_k = np.array([[4.721,0.143,0.006], # k=1
                    [9.353,0.284,0.007], # k=2
                    [14.773,0.422,0.009], # k=3
                    [18.246,0.555,0.011], # k=4
                    [22.609,0.715,0.012],
                    [22.609,0.715,1.582],
                    [22.609,3.915,1.512],
                    [22.609,7.515,1.582],

```

```

[22.609,11.405,1.582],
[22.945,15.715,1.582],
[22.122,18.955,1.482],
[22.609,22.715,1.501],
[22.609,22.715,1.582],
[22.609,22.715,0.012],
[26.609,22.715,0.012],
[30.109,22.715,0.012],
[34.609,22.715,0.012],
[38.609,22.715,0.012]])#k=5
'''
z_k = np.array([-0.038814776,0.665418647,0.028274334],# k=2
               [-0.086377027,1.331266279,0.05131268], # k=3
               [-0.146457194,1.998432625,0.06981317], # k=4
               [-0.151833438,2.004551048,0.396015207],
               [-0.459243719,2.598858927,0.437379511],
               [-0.791128366,3.177177062,0.481012742],
               [-0.791626073,3.174910566,0.130899694],
               [-0.79321195,3.173223397,5.974436563],
               [-0.616398744,3.814517234,5.994158783],
               [-0.416947898,4.446596753,5.987526532],
               [-0.275393488,5.10412318,6.031159763],
               [-0.157973032,5.764025631,6.067113546],
               [-0.15596121,5.766868647,4.297873283],
               [-0.154549699,5.770067457,3.511427922],
               [-0.156681123,5.765952698,3.086963848],
               [-0.156347772,5.092435617,3.112096589],
               [-0.1599123,4.427322629,3.126233756],
               [-0.113302429,3.757513028,3.171088718],
               [-0.052256155,3.089100232,3.202679177],
               [-0.040383233,3.077983959,3.581241092],
               [-0.031588472,3.075026369,3.987030143],
               [0.50886064,2.683984757,4.036073895],
               [0.508750134,2.685139324,3.637964293],
               [0.509073608,2.68625966,3.248406804],
               [0.63279035,2.033419252,3.288898442],
               [0.771736173,1.377950803,3.320488902],
               [0.952927236,0.734739754,3.366216528],
               [1.129476021,0.090965133,3.389254874],
               [1.129982374,0.086338541,1.749692575],
               [0.480357415,-0.072964494,1.781283035],
               [0.4710994,-0.071961933,0.052359878]])#k=5
'''

# The estimated state vector at time k-1 in the global reference
frame.
# [x_k_minus_1, y_k_minus_1, yaw_k_minus_1]
# [meters, meters, radians]

```

```

state_estimate_k_minus_1 = np.array([0.0,0.0,0.0])

# The control input vector at time k-1 in the global reference
frame.
# [v, yaw_rate]
# [meters/second, radians/second]
# In the literature, this is commonly u.
# Because there is no angular velocity and the robot begins at
the
# origin with a 0 radians yaw angle, this robot is traveling
along
# the positive x-axis in the global reference frame.
control_vector_k_minus_1 = np.array([4.0,0.0])

# State covariance matrix P_k_minus_1
# This matrix has the same number of rows (and columns) as the
# number of states (i.e., 3x3 matrix). P is sometimes referred
# to as Sigma in the literature. It represents an estimate of
# the accuracy of the state estimate at time k made using the
# state transition matrix. We start off with guessed values.
P_k_minus_1 = np.array([[0.15,  0,   0],
                        [ 0,0.15,   0],
                        [ 0,  0,  0.15]])

# Start at k=1 and go through each of the 5 sensor observations,
# one at a time.
# We stop right after timestep k=5 (i.e., the last sensor
observation)
for k, obs_vector_z_k in enumerate(z_k,start=1):

    # Print the current timestep
    print(f'Timestep k={k}')

    # Run the Extended Kalman Filter and store the
    # near-optimal state and covariance estimates
    optimal_state_estimate_k, covariance_estimate_k = ekf(
        obs_vector_z_k, # Most recent sensor measurement
        state_estimate_k_minus_1, # Our most recent estimate of
the state
        control_vector_k_minus_1, # Our most recent control
input
        P_k_minus_1, # Our most recent state covariance matrix
        dk) # Time interval

    # Get ready for the next timestep by updating the variable
values
    state_estimate_k_minus_1 = optimal_state_estimate_k

```

```
P_k_minus_1 = covariance_estimate_k

# Print a blank line
print()

# Program starts running here with the main method
main()
plt.show()
```