SMART WHEELCHAIR(TWO-WHEELED) ACTUATION USING ROS (ROBOTIC OPERATING SYSTEM)

A TERM PROJECT REPORT

Submitted by

AKASH A (CB.EN.P2EBS24001)

AKSHARA ARUN(CB.EN.P2EBS24002)

KEERTHANA M G (CB.EN.P2EBS24011)

MUTHU KRISHNAN S (CB.EN.P2EBS24015)

In partial fulfillment for the award of the degree of

MASTER OF TECHNOLOGY
IN
EMBEDDED SYSTEMS

DEPARTMENT OF ELECTRICAL AND ELECTRONICS

ENGINEERING

AMRITA SCHOOL OF ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITANAGAR, COIMBATORE - 641 112

APRIL 2025

ABSTRACT

This paper introduces the design and development of a smart wheelchair prototype from a two-wheeled differential drive robot using the Robot Operating System 2 (ROS 2) Humble distribution on a Raspberry Pi 4. The main goal is to provide real-time motion control of the robot using direct wheel actuation, without the need for any external sensors. The system architecture is minimal to test core mobility features, using only GPIO-based motor control.

User commands are input using keyboard teleoperation through the teleop_twist_keyboard ROS 2 package. These commands are published in the form of Twist messages to the /cmd_vel topic, which contains both linear and angular velocity data. A custom ROS 2 subscriber node decodes these messages and translates them to digital signals for motor control through the L298N motor driver module. This allows for directional motion such as forward, reverse, and rotating in place, exhibiting rudimentary robotic locomotion through ROS 2 message transmission.

The simplicity of the system is also a proof of concept for motor actuation based on ROS 2 and low-cost hardware. It provides a foundation for future upgrade capabilities like sensor-based obstacle avoidance, self-navigating, and intelligent decision-making. The suggested model provides a flexible and scalable platform to create assistive mobility solutions and is also an educational aid to develop robotics and embedded systems.

INDEX

TABLE OF CONTENT

CHAPTER -1
INTRODUCTION

The creation of assistive robotic systems, like smart wheelchairs, has the goal to enhance mobility and autonomy for disabled people. Although most smart wheelchair systems integrate multiple sensors (e.g., ultrasonic, LiDAR, or cameras) for navigation and obstacle avoidance, such complexity will lead to higher cost, power consumption, and time-to-market.

This project introduces a simple smart wheelchair prototype realized as a two-wheeled differential drive robot with a sole emphasis on wheel actuation without the utilization of any sensors. The system is developed based on the Robot Operating System 2 (ROS 2) Humble on a Raspberry Pi 4, selected due to its flexibility, open-source community support, and embedded platform compatibility.

Movement commands are issued through keyboard teleoperation with the teleop_twist_keyboard ROS 2 package. These commands are published to the default /cmd_vel topic as Twist messages with linear and angular velocity values. A dedicated ROS 2 subscriber node on the Raspberry Pi translates these messages and produces corresponding GPIO outputs. These inputs are sent to an L298N motor driver, which in turn drives two DC motors to move the wheels.

The system receives basic directional commands: forward, backward, left, right, and stop. Even without sensor feedback, the implementation proves successful in showcasing a low-cost, lightweight, and modular actuation system with ROS 2, making it ideal for initial-stage prototyping and educational use.

Such base configuration also acts as an initiating platform to implement future extension ideas, such as sensor integration, voice guidance, or self-governing drives, over a period, fully developing into an end-to-end smart mobility system.

CHAPTER -2
LITERATURE SURVEY

2.1 Semi-Autonomous Teleoperated Robot for Enhanced Human-Machine Interaction

Conference/Journal: IEEE Access conference, 2025

This paper explains the design of a teleoperated, semi-autonomous robot intended to improve human-machine interaction by implementing sophisticated technologies like the Internet of Things (IoT) and Robot Operating System 2 (ROS 2). The robot is developed on a Raspberry Pi 4 using the ROS 2 Humble framework to offer stable and accurate control. It has a differential drive system, enabling it to move smoothly on different terrain. The teleoperation is enhanced by an optimized teleop-key algorithm, resulting in a more responsive and controllable robot when operated remotely. IoT integration is important as it facilitates real-time monitoring, remote access, and data-driven decision-making. The connectivity improves the robot's adaptability, operational safety, and efficiency. The integration of ROS 2 and IoT makes the robot very efficient for field operations like exploration, environmental monitoring, and sophisticated operational tasks, demonstrating the capability of these technologies to push the boundaries of field robotics.

2.2 A Framework for Remote Robot Actuation using ROS Integrated with MQTT

Conference/Journal: IEEE Access conference, 2024

Conventionally, robots operating with the Robot Operating System (ROS) could only be commanded from the same machine. This paper suggests a new paradigm that eliminates this limitation by combining ROS with the MQTT protocol to enable remote control of robots on any device that accesses the internet. To test this configuration, the authors constructed an operational prototype and conducted tests in two scenarios: local control (from the same device) and remote control (over the internet). The findings indicated that despite a minimal delay (latency) when controlling the robot from a distance, the robot nonetheless carried out all commands perfectly with 100% accuracy. Nonetheless, given the internet-delay, alternative methods may prove better for work requiring very high levels of precise and time-based movement

2.3 Internet of Things and Artificial Intelligence Enabled Smart Wheel Chair

Conference/Journal: IEEE Access conference, 2023

This paper introduces an IoT-based smart wheelchair that is intended to assist disabled and elderly people by tracking health metrics such as temperature, blood pressure, and fall detection through sensors interfaced with a Raspberry Pi 4. It has an emergency alert system that informs caretakers through a mobile application, allowing them to respond through an Alexa-enabled device with a camera and monitor. Also, there is a hydraulic mechanism that aids in sitting and standing, hence the system proving to be effective and efficient care and mobility solution.

2.4 Smart Voice and Gesture Controlled Wheel Chair

Conference/Journal: IEEE Access conference, 2022

This paper introduces a cost-efficient smart wheelchair that utilizes voice and gesture control to assist physically disabled persons. Conventional smart wheelchairs tend to be costly and large, thereby hard to attain and maneuver. In order to solve these challenges, the system to be developed employs a VR3 voice recognition module and Arduino UNO to decode commands like LEFT, RIGHT, FORWARD, and BACKWARD that can be issued using voice or small movements with the hand, finger, elbow, or head. The system should be lightweight, user-friendly, and simple to use so that users can navigate independently with little physical effort. Even with the potential advantages, this kind of technology is underutilized in local markets because of hardware and cost limitations. Nevertheless, the successful prototype development proves its usability and worth in enhancing mobility and independence for the disabled population.

2.5 ADRC Attitude Controller Based on ROS for a Two-Wheeled Self-Balancing Mobile Robot

Conference/Journal: IEEE Access Journal, 2023

Here, the authors discuss a ROS-based control mechanism of a two-wheeled self-balancing robot based on the Active Disturbance Rejection Control (ADRC) approach. The robot, developed for educational applications, is equipped with Raspberry Pi 4 and Arduino Mega for high and low-level processing, respectively. The control mechanism employs MPU6050 sensors and DC motors with encoders. The ADRC approach along with an Extended State Observer (ESO) aids in improving stability by accounting for disturbances and uncertainties. Experimental outcomes verify the system's effectiveness, simplicity in implementation, and real-time control using minimal hardware resources.

2.6 ROS-Based Indoor Autonomous Exploration and Navigation Wheelchair

Conference/Journal: IEEE Access Conference, 2017

The present paper introduces an affordable and recyclable indoor autonomous navigation system with the sole intention of being deployed in smart wheelchairs. For minimizing hardware costs and complexity, the system uses an Arduino microcontroller to drive the wheelchair's motors using a two-wheel differential drive. The high-level operations are performed using the Robot Operating System (ROS) in order to enable modular and scalable software integration. In contrast to costly laser sensors, the system utilizes a low-cost RGB-D camera, which provides both color and depth data. The depth information is converted into laser-like inputs to support environment mapping and obstacle detection based on the Gmapping algorithm. Localization is performed using the Adaptive Monte Carlo Localization (AMCL) algorithm to find the wheelchair's correct position and orientation. For navigation, the A* algorithm produces optimal global paths and the Dynamic Window Approach (DWA) provides smooth local path modifications and real-time obstacle avoidance. The system also utilizes a frontier-based exploration algorithm, enabling the wheelchair to automatically find and map unknown spaces. Moreover, an Android app offers an intuitive interface for user control and interaction. In summary, the system presented here showcases a cost-effective and viable solution for smart wheelchair autonomous navigation for use in real-world applications of healthcare and assistive robotics.

2.7 ROS Based Control of Robot Using Voice Recognition

Conference/Journal: IEEE Access Conference, 2019

The research aims at the design of a voice-operated wheeled robot based on the Robot Operating System (ROS) platform. The robot is constructed with a differential drive system, through which it can travel and turn based on the speed control of two wheels separately. The novelty of the system is that it is designed to comprehend and carry out verbal instructions given by the user. This is done through a voice recognition system using Hidden Markov Models (HMM) — a statistical approach to modeling speech patterns. The Viterbi algorithm, which is widely applied in speech signal decoding, assists in improving recognition accuracy by identifying the most likely sequence of hidden states (i.e., words or phonemes). After the voice command has been identified, the robot performs the corresponding movement (such as forward, backward, left, or right). This system illustrates how voice interfaces can simplify robot operation, particularly in smart home or assistive technology use.

2.8 ROS-FM: Fast Monitoring for the Robotic Operating System(ROS)

Conference/Journal: IEEE Access Conference, 2020

This paper presents ROS-FM, a high-throughput, inline network-monitoring framework for both ROS1 and ROS2 platforms. It takes advantage of eBPF and XDP technologies to monitor network traffic in real-time with little overhead. ROS-FM is complemented with two important elements: a security policy enforcement component to identify and block unauthorized operations, and a distributed data visualization component for real-time network awareness across robotic platforms.

By conducting experiments, the authors compare ROS-FM to other traditional ROS monitoring tools and discover that ROS-FM far outperforms them, particularly in busy environments with over 10 active robot processes. Incredibly, in an environment with 80 processes, ROS-FM only consumes 4% of the overhead of generic tools. Furthermore, the security tool in ROS-FM is extremely effective in fending off typical network-based attacks in both ROS1 and ROS2 environments. This renders ROS-FM as light, secure, and scalable robotic system monitoring solution.

CHAPTER-3
OBJECTIVE

The primary aim of this project is to implement and design a smart wheelchair prototype based on a two-wheeled differential drive system with actuation controlled using ROS 2 Humble over a Raspberry Pi 4. The project specifically deals with wheel actuation and fundamental movement control, without employing any sensors within the system. The following are the specific objectives in detail:

3.1 Design and Implement Wheel Actuation System:

- Create a system where the movement of the wheelchair is governed by motor actuation using a differential drive system, thus allowing smooth control of the robot's wheels.
- Use ROS 2 Humble to execute the commands for the movement of the robot to achieve scalability and modularity for future development.

3.2 Support Keyboard-based Teleoperation:

- Integrate the teleop_twist_keyboard package in ROS 2 in order to enable the user to control the wheelchair's movement remotely from the keyboard.
- Map the keyboard commands to ROS 2's Twist messages with linear velocity and angular velocity data, which will be utilized to manipulate the speed and direction of the robot.

3.3 Translation of Commands to Motor Control:

- Create a ROS 2 subscriber node that accepts the Twist messages published on the /cmd_vel topic and translates the commands into GPIO signals.
- Use these GPIO signals to power the motors via an L298N motor driver to provide proper motor direction and speed according to the velocity commands.

3.4 Basic Mobility Functions:

- Implement basic direction of movements using forward, backward, left and right turns, and stop. These are implemented using the differential drive model where left and right motors are independently controlled.

- This gives simple manual wheelchair control, and the user will have complete authority over its motion.

3.5 Provide a Platform for Future Upgrades:

- This prototype system serves as an initial foundation on which sensor-based feedback (i.e., obstacles, distance measurements) and automatic navigation can be added in the future.
- Subsequent upgrades can include voice recognition for simpler usage or compatibility with other assistive devices to help improve mobility and independence.

3.6 Low-Cost and Modular Design:

- The platform is made to be low-cost, utilizing readily available parts like a Raspberry Pi 4 and an L298N motor driver. This makes the platform perfect for prototyping and educational use.
- The system's modular design provides flexibility, which includes future upgrades, like the inclusion of sensors, new communication interfaces, or AI-based decision-making.

CHAPTER-4
HARDWARE/SOFTWARE SPECIFICATION

4.1 HARDWARE SPECIFICATION:

4.1.1 Raspberry Pi 4

- Runs ROS 2 Humble on Ubuntu 22.04
- 40 GPIO pins for motor control
- Powered via 5V 3A USB-C supply

4.1.2 L298N Motor Driver Module

- Dual H-Bridge, supports 5V–35V motors
- Drives two 12V DC motors using GPIO signals
- Max 2A per channel

4.1.3 12V DC Motors (x2)

- Generic brushed motors
- Connected to wheels for differential drive
- Powered via L298N driver

4.1.4 Power Supply

- 12V 2A adapter
- Powers Raspberry Pi and motor components

4.1.5Motor Wheels

- 6-inch rubber wheels
- Mounted directly to motors for mobility

4.1.6 Wiring & Connectors

- Jumper wires for GPIO connections

- Motor leads and power cables for clean layout

4.1.7Other Components

- USB Keyboard for teleoperation (via teleop_twist_keyboard)
- Two-wheeled robot chassis for mounting all parts

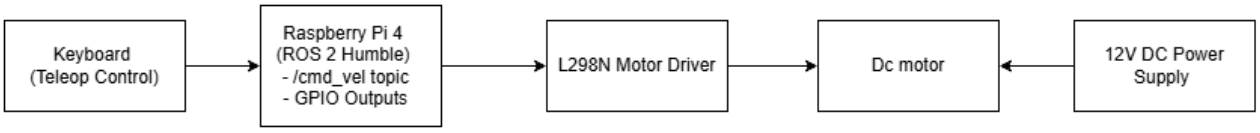


Figure 4.1 Hardware Architecture

4.2 SOFTWARE SPECIFICATION:

1. Operating System

- **Ubuntu 22.04 LTS (64-bit)**
- Lightweight and stable OS suitable for ROS 2 Humble.

2. ROS 2 Distribution

- **ROS 2 Humble**
- Middleware used for real-time robotic communication.
- Key packages used:
  - teleop_twist_keyboard – For keyboard-based velocity control.
  - rclpy – Python client library for ROS 2 node development.
  - geometry_msgs/Twist – For velocity message types (/cmd_vel topic).

3. Programming Language

- **Python 3**
- Used to create ROS 2 nodes for GPIO motor control.

4. GPIO Control Library

- **RPi.GPIO** or **gpiozero**
- Enables access to Raspberry Pi's GPIO pins for motor control via code.

5. Terminal Tools

- **GNOME Terminal / LXTerminal**

- Used to launch ROS nodes and execute teleoperation scripts.

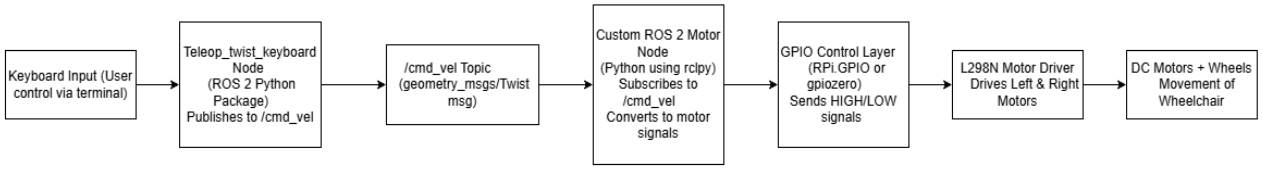


Figure 4.2 Softwareware Architecture

CHAPTER-5
METHODOLOGY

5.1 Overview of ROS 2 in the Project

ROS 2 (Robot Operating System) is a modular and flexible middleware framework to build robot applications. It implements a publish-subscribe model, where nodes exchange information through topics, services, and actions. In this project, ROS 2 Humble serves as the central system to handle real-time communication between actuation (motors) and input (keyboard).

5.2 ROS Architecture:

A ROS 2 workspace dedicated (a directory structure) is used to hold all smart wheelchair-related packages.

Inside the workspace, source code, launch files, configuration files, and custom messages (if necessary) are structured for improved modularity.

A DIRECTORY STRUCTURE

```
ros2_ws/
├── src/
│   └── two_wheel_robot/
│       ├── launch/
│       │   └── two_wheel_launch.py
│       ├── two_wheel_robot/
│       │   ├── __init__.py
│       │   ├── teleop_node.py(NODE1)
│       │   └── motor_driver_node.py(NODE2)
│       ├── package.xml
│       └── setup.py
```

5.3 ROS 2 Node Overview

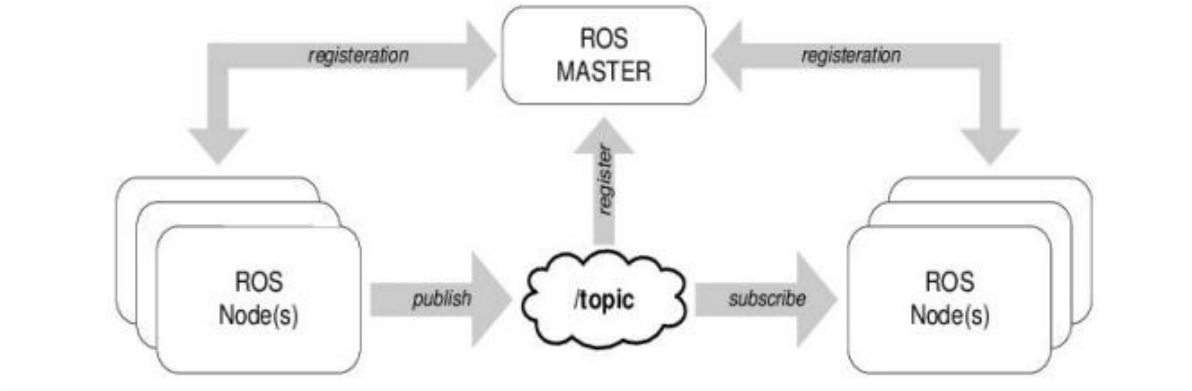


Figure5.1 ROS publish-subscribe communication model

5.3.1 Node 1: Keyboard Teleoperation Node

- This node is responsible for capturing keyboard inputs.
- It maps key presses to velocity commands (linear and angular velocity).
- These commands are then published to a topic named **/cmd_vel**.
- It uses an existing ROS 2 package: teleop_twist_keyboard.

5.3.2 Node 2: Motor Control Node

- This node subscribes to the **/cmd_vel** topic.
- It receives velocity data and translates it into **motor direction and speed** signals.
- These signals are converted to digital outputs using the **GPIO** pins of the Raspberry Pi.
- The outputs are fed into the **L298N motor driver** to drive the motors.

5.4 Node-to-Node Information Flow

- **User Interaction**: User enters arrow keys or WASD on the keyboard.
- **Teleoperation Node**: Detects the keypress, translates it into a velocity message.
- **ROS 2 Topic (/cmd_vel):** Shared communication channel.
- **Motor Control Node**: Listens for /cmd_vel, translates the message.
- **GPIO Signal Generation**: Depending on translated velocity, GPIO pins are driven HIGH or LOW.
- **Motor Driver (L298N):** Takes the GPIO signals and drives the motors.
- **Wheel Movement:** Motors spin wheels in the desired direction.

5.5. Node Communication Model

- **Topic Name**: /cmd_vel
- **Message Type**: geometry_msgs/msg/Twist
- **Publisher Node**: teleop_twist_keyboard
- **Subscriber Node**: motor_controller_node
- **Communication Type**: Asynchronous via DDS (Data Distribution Service)

5.6 Launching the System

- A **ROS 2 launch file** is created to start both nodes together.
- When the system is launched:
    - Teleoperation node starts first to read keyboard inputs.
    - Motor controller node initializes to listen for commands.
    - Upon input, real-time wheel movement is achieved.

5.7 Movement Mechanism Using Differential Drive

The robot is based on a **two-wheeled differential drive** mechanism:

- **Forward**: Both motors move forward.
- **Backward**: Both motors reverse.
- **Left Turn**: Left motor moves backward, right moves forward.

- **Right Turn**: Left motor forward, right backward.
- These movements are purely controlled via velocity command interpretation.

5.8 Key Features of the ROS 2 Framework Used

- **Modularity**: Each function (input, control, movement) is separated into nodes.
- **Real-Time Capability**: ROS 2 supports real-time data handling through DDS.
- **Scalability**: New features (like sensors, maps) can be added by creating more nodes or modifying existing ones.
- **Portability**: The same setup can be transferred to any ROS-supported hardware.

5.9 Summary of ROS Nodes and Their Roles

Node Name	Role	Communication
teleop_twist_keyboard	Captures keyboard input and publishes velocity	Publishes to /cmd_vel
motor_controller_node	Reads velocity and actuates wheels via GPIO	Subscribes to /cmd_vel

CHAPTER-6
RESULT

The implementation of the smart wheelchair prototype using ROS 2 Humble on a Raspberry Pi 4 successfully achieved the goal of two-wheeled differential drive actuation without incorporating any sensors. The system demonstrated reliable, real-time control of the wheelchair's motion through keyboard teleoperation.

6.1 Keyboard Teleoperation: The teleop_twist_keyboard node accurately captured user input and published velocity commands (/cmd_vel) corresponding to desired movement directions (forward, backward, left, and right).

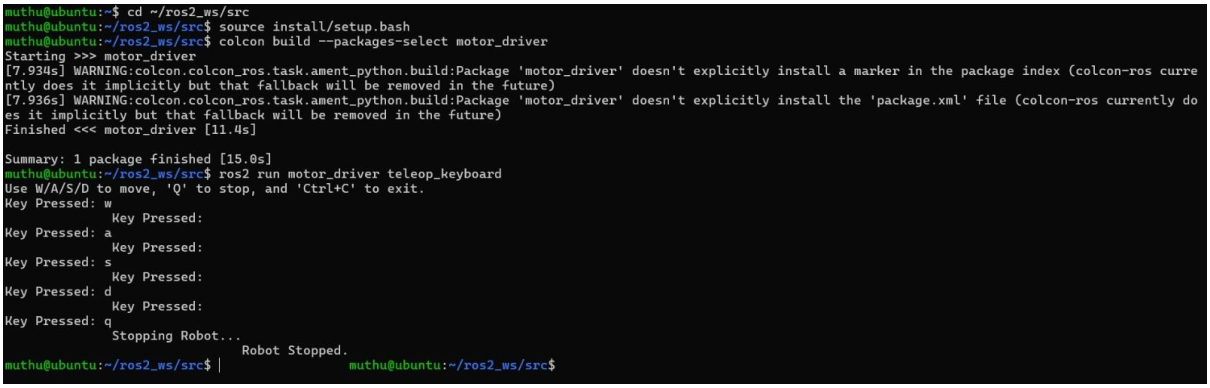


Figure 6.1 Node1:teleop node

6.2 Motor Actuation via GPIO: The motor control node successfully received the velocity commands and translated them into appropriate GPIO signals. These signals were used to drive the L298N motor driver, enabling directional movement of the two DC motors.

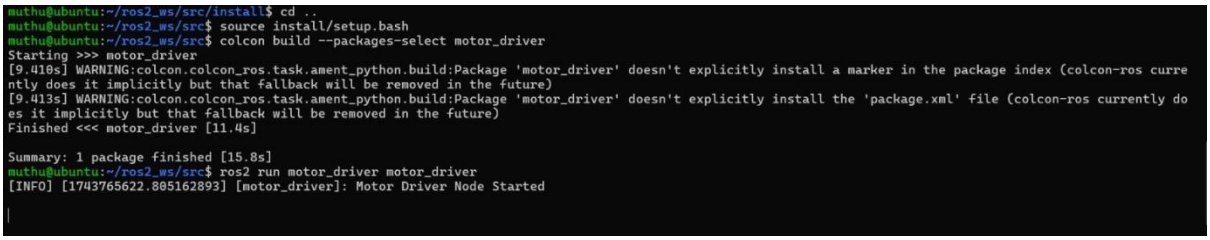


Figure 6.2 Node2:Motor driver Node

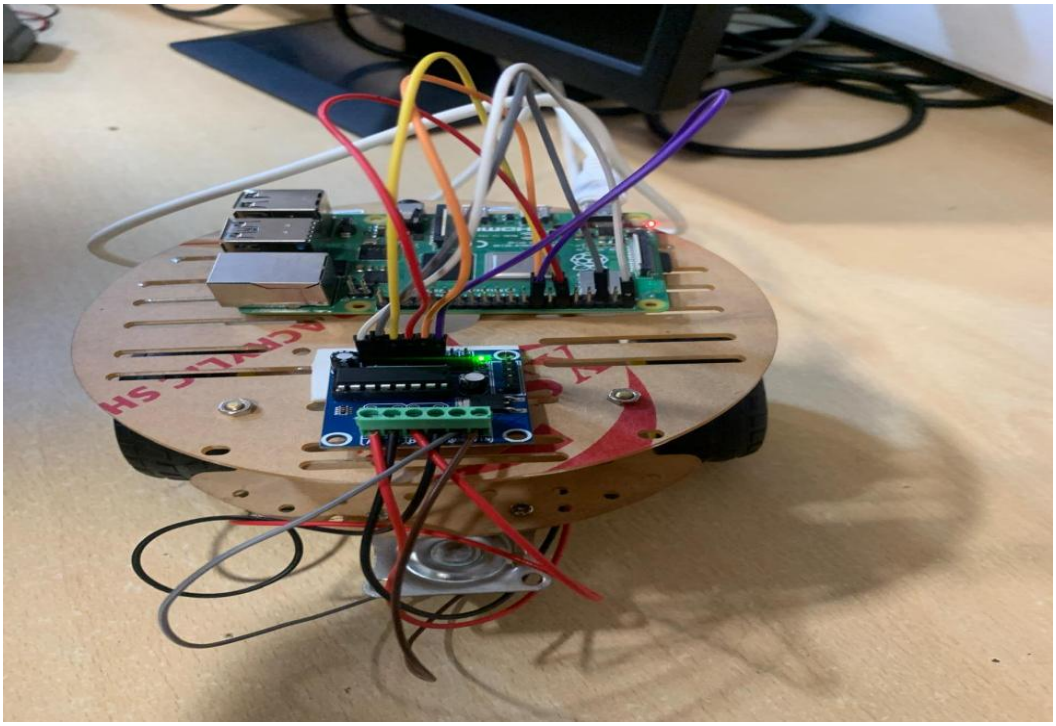
6.3 Real-Time Responsiveness: The system exhibited minimal delay between input and motion, demonstrating the efficiency of ROS 2's DDS-based communication in real-time applications.

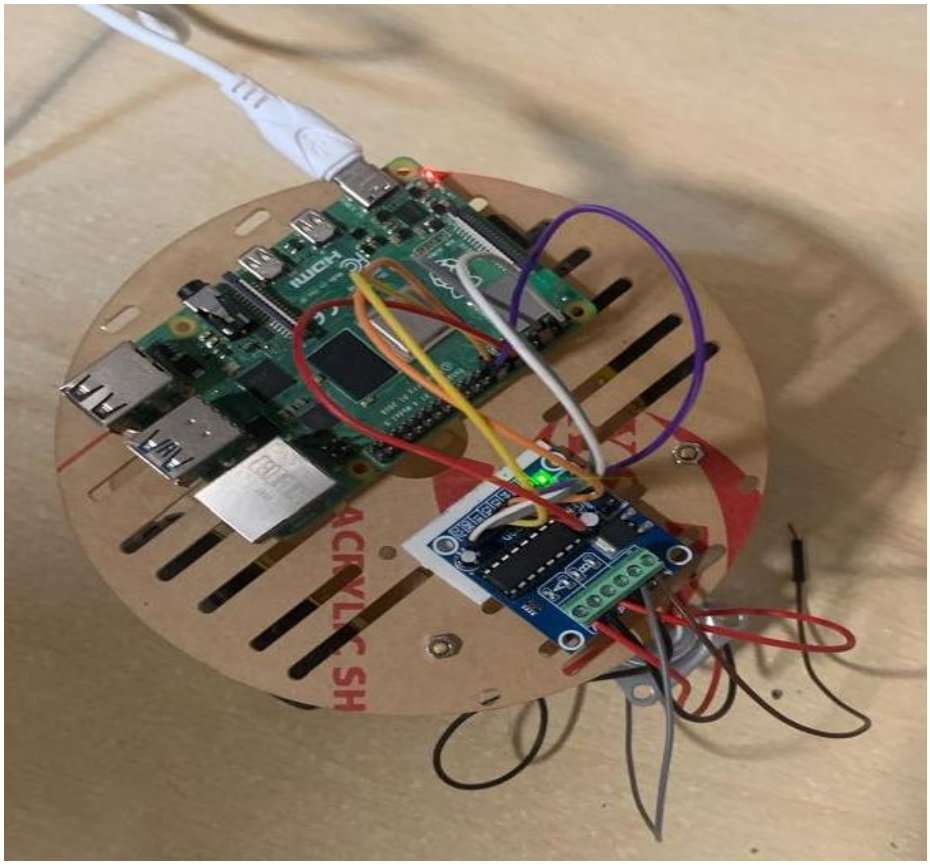
6.4 Stable Movement: The differential drive system provided smooth and stable motion on flat surfaces, confirming the effectiveness of the hardware-software integration.

6.5Modular and Scalable Design: The modular node-based architecture allows for seamless future enhancements, such as integration of sensors, automation, or voice control.

6.6 Visual Proof of Concept:

- The robot responded to W, A, S, D key inputs to move forward, left, backward, and right respectively.
- Direction changes were smooth and consistent with no noticeable lag or miscommunication between nodes.





CHAPTER-7
FUTURE ENHANCEMENTS

- **Sensor Integration**: Add ultrasonic, LiDAR, or IMU sensors for obstacle detection and environment awareness.
- **Autonomous Navigation**: Use ROS 2 navigation stack and SLAM for path planning and self-driving features.
- **Voice/Gesture Control**: Enable hands-free control using speech or hand gestures.
- **Mobile App**: Create an app for remote control and system monitoring.
- **Health Monitoring**: Integrate vital sensors for user safety.
- **Safety Features**: Add fall detection and emergency stop functions.
- **Chassis Upgrade**: Improve comfort, structure, and usability for daily use.

CHAPTER-8
CONCLUSION

This project effectively validates the fundamental actuation of a two-wheeled intelligent wheelchair employing ROS 2 Humble on a Raspberry Pi 4 without the use of any external sensor. The use of the teleop_twist_keyboard package permitted real-time control via keyboard inputs, while ROS 2's communication layer allowed for smooth command delivery to the motors through GPIO and an L298N motor driver.

The system succeeded in having reproducible differential drive motion, making it a building block for enhanced capabilities like collision detection, speech control, and autonomous way-finding. Due to the modularity of the ROS 2 structure and low-level hardware interface, the prototype is economical, expandable, and perfect for use in pedagogic, research, or assistive mobility applications.

By and large, this contribution constitutes a promising milestone toward realizing accessible and intelligent robotic mobility solutions.

CHAPTER-9
REFERENCE

[1]G. A. Muñoz-Hernandez, J. Díaz-Téllez, J. Estevez-Carreon and R. S. García-Ramírez, "ADRC Attitude Controller Based on ROS for a Two-Wheeled Self-Balancing Mobile Robot," in IEEE Access, vol. 11, pp. 94636-94646, 2023, doi: 10.1109/ACCESS.2023.3308948.keywords: {Robots;Mobile robots;Observers;Attitude control;Mathematical models;Heuristic algorithms;DC motors;ADRC;ESO;mobile robot;ROS;two-wheeled self-balancing;segway},

[2] C. Lertyosbordin, D. Wongsanont, N. Khurukitwanit and W. Saowapark, "A Framework for Remote Robot Actuation using ROS Integrated with MQTT," 2024 International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC), Okinawa, Japan, 2024, pp. 1-4, doi: 10.1109/ITC-CSCC62988.2024.10628235. keywords: {Motor drives;Actuators;Protocols;Operating systems;Prototypes;Publish-subscribe;Real-time systems;Control;MQTT;Remote;Robot;ROS},

[3] G. B. Menti, V. R. Chowdary, N. L. P. Nagasarapu, G. K. Godugunuri, P. Abburi and J. Yerramsetty, "Semi-Autonomous Teleoperated Robot for Enhanced Human-Machine Interaction," 2025 International Conference on Multi-Agent Systems for Collaborative Intelligence (ICMSCI), Erode, India, 2025, pp. 581-586, doi: 10.1109/ICMSCI62561.2025.10894321. keywords: {Human computer interaction;Simultaneous localization and mapping;Accuracy;Navigation;Robot vision systems;Cameras;Real-time systems;Agriculture;Safety;Internet of Things;Semi-Autonomous system;ROS 2 Humble;IoT;Teleop-key;Human-Machine Interaction;Rpi 4},

[4] Z. Li, Y. Xiong and L. Zhou, "ROS-Based Indoor Autonomous Exploration and Navigation Wheelchair," 2017 10th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 2017, pp. 132-135, doi: 10.1109/ISCID.2017.55. keywords: {Wheelchairs;Navigation;Cameras;Algorithm design and analysis;Smart phones;Image edge detection;Heuristic algorithms;intelligent wheelchair;autonomous exploration;ROS;simultaneous localization and mapping (SLAM);human-computer interaction},

[5] S. Rivera, A. K. Iannillo, S. Lagraa, C. Joly and R. State, "ROS-FM: Fast Monitoring for the Robotic Operating System(ROS)," 2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS), Singapore, 2020, pp. 187-196, doi: 10.1109/ICECCS51672.2020.00029. keywords: {Data visualization;Tools;Monitoring;Robots;Penetration testing;eBPF;XDP;ROS;Security},

[6] R. K. Megalingam, R. S. Reddy, Y. Jahnavi and M. Motheram, "ROS Based Control of Robot Using Voice Recognition," 2019 Third International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 2019, pp. 501-507, doi: 10.1109/ICISC44355.2019.9036443. keywords: {Hidden Markov models;Speech recognition;Software;DC motors;Robot kinematics;Mobile robots;voice-controlled robot;Pocket Sphinx;voice recognition;ROS},

[7] K. L. Narayanan, R. Niranjana, S. Vinothini, R. S. Krishnan, G. V. Rajkumar and L. Rachel, "Internet of Things and Artificial Intelligence Enabled Smart Wheel Chair," 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 2023, pp. 1436-1440, doi: 10.1109/ICICT57646.2023.10134156. keywords: {Temperature sensors;Temperature measurement;Wheelchairs;Virtual assistants;Wheels;Hydraulic systems;Control systems;Smart Wheelchair;Internet of Things;Alexa;Raspberry pi;Sensors;Hydraulics},

[8] R. Khande and S. Rajapurkar, "Smart Voice and Gesture Controlled Wheel Chair," 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2022, pp. 413-417, doi: 10.1109/ICOEI53556.2022.9777223. keywords: {Image recognition;Wheelchairs;Sociology;Wheels;Prototypes;Machine learning;Speech recognition;Gesture recognition;voice recognition;gesture control;microcontroller;wheel chair;Arduino},