

**Arab Academy for Science, Technology and Maritime Transport**

**College of Engineering and Technology**

**Computer Engineering**

B. Sc. Final Year Project

**Voice Control Wheeling Chair**

**(Proof Of Concept)**

*Presented By:*

*Abdelrahman Mohamed Radwan  
Marwan Mohamed  
Mohamed Elsayed*

*Mohab Mohamed Bahr   
Mohamed Ehab Hamed  
Mohamed Ahmed Moghazy*

*Supervised By:*

*Prof.Ossama Ismail - Prof. Walid Ghoneim*

**Declaration**

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Bachelor of Science in *(Computer and Mechatronics Engineering)* is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

**Signed:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Registration No.:** \_\_\_\_\_\_\_\_\_\_\_

**Date:**

****

**Arab Academy for Science, Technology and Maritime Transport**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**Department of Computer Engineering**

Academic Year: 2023/2024 Semester: Spring

**Senior Project Summary Report**

|  |  |  |
| --- | --- | --- |
| **Project Title** | ***Voice Control Wheeling Chair*** | |
| **Supervisor(s)** | **Prof. Walid Ghoneim**  **Prof. Osama Ismail** | |
| **Team members:** | **Mohamed Moghazy**  **Mohamed Ehab**  **Mohamed Elsayed**  **Mohab Bahr**  **Marwan Mohamed**  **Abdelrahman Radwan** | **19100650**  **19101839**  **19101997**  **19102769**  **19100983**  **19100479** |
| **Project Deliverables** | **Voice control chair for disabled people to help them to reduse their suffering in living normal life and try making them depend on their self.** | |
| **Abstract** | This paper presents a proof of concept for a new voice-controlled wheelchair system that integrates mechanical design, voice recognition, and computer vision technologies. The proposed solution combines advanced technologies to design an intuitive and efficient mobility solution for people with physical disabilities. The mechanical design ensures a stable and comfortable riding experience by considering user comfort, safety, material selection, ergonomic factors, and structural integrity. The voice recognition module employs advanced algorithms and natural language processing techniques to accurately recognize and respond to spoken commands, enabling hands-free control for navigation, speed adjustment, and other functions. The computer vision system enhances environmental awareness and navigation capabilities through real-time camera feed analysis, allowing the wheelchair to identify obstacles and navigate complex environments safely and autonomously.  The proof of concept involves experimental testing within controlled settings to evaluate the functionality and performance of the integrated system. Initial results are encouraging, demonstrating promising usability and effectiveness, and laying the groundwork for further development and refinement. Future research will focus on enhancing the mechanical design for portability, improving the interaction quality and responsiveness of the voice recognition and computer vision algorithms, and evaluating the system's performance in real-world scenarios. This proof of concept represents a significant step toward developing novel voice-controlled wheelchair systems that ensure mobility and independence for individuals with mobility impairments. | |
| **Engineering Standards** | **IEEE 802.11 (Wi-Fi)**: System uses wireless communication.  **IEEE 12207**: For software life cycle processes.  **IEC 60601-1**: Medical electrical equipment—Part 1: General requirements for basic safety and essential performance.  **IEC 61508**: Functional safety of electrical/electronic/programmable electronic safety-related systems.  I**SO/IEC 12207**: Systems and software engineering—Software life cycle processes.  **ISO/IEC 25010**: Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models.  **ISO 7176-1**: Wheelchairs—Part 1: Determination of static stability.  **ISO 7176-2**: Wheelchairs—Part 2: Determination of dynamic stability of electric wheelchairs.  **ISO 7176-3**: Wheelchairs—Part 3: Determination of the effectiveness of brakes. | |
| **Design Constraints** | **1- Price : price of lightweight high- strength materials anf their machining cost is signeficently high**  **2- Labor Experience : we don't have the machines nor the labor that can work on new materials like composite metals or very-high grade metals or even polymers**  **3- Construction : The construction should withstand minor and major impacts due to motion Ex: dropping a step**  **4- Control : We have a problem merging 2 AI models working simultanously with only 4GB of RAM while the time needed to further inspect the improve the Models themselves is huge** | |
| **Project Impact** | **Social : Trying to co-op Special Need human-beings again into community without making them feel they are a burden on their people** | |
| **Team Organization** | **Marwan Mohamed : maintaining the sensory fusion and merging processes between multiple different system in order to work coherently**  **- Mohamed El Sayed : Plotting the main points needed in order to make the whole system works flawlessly and interact with each part of the system to make sure everything is intact**  **Mohamed Moghazy :testing and applying voice code on raspberry pi**  **Abdelrahman radwan: searching for best modules**  **Mohamed Ehab :responsible for camera detection**  **Mohab Bahr: training the models and measuring distance** | |
| **Ethics /Safety** | **1- Totally Air-bubbled : It has to be controlled from the inside only in order to protect the user's privacy from being located at every time he/she uses it**  **2- Cost Minimization : It can introduced to non-profitable orginization or even a company but not all user can withstand the price of this technology being newly introduced to the market so cost has to be minimum while maintaining good product**  **3- Material Cutdown : We had to balance between safety of the wheelchair itself and being light weighted but also cheap as possible keeping in mind to be robust enough to survive every user)** | |

**Main Supervisor Signature**

**……………………………………..**

To all those who have inspired and supported us,

This voice control wheeling chair project is dedicated to you

**Acknowledgment**

We are incredibly grateful to everyone who made the development of our voice-controlled wheelchair a reality.

First and foremost, I want to express my deepest gratitude to our project supervisor, Prof.Waleed Ghoneim,Prof.Osama Ismail. Your unwavering support, insightful guidance, and constructive feedback were instrumental in navigating the complexities of this project. Thank you for believing in our vision and helping us bring it to life.

A big thank you for providing us with the resources and facilities necessary to pursue this project. The technical staff, in particular, deserve special mention for their prompt assistance and expertise, which were crucial during the setup and testing phases.

We would also like to extend my sincere thanks to the amazing team . Your collaborative spirit and willingness to share your knowledge in voice recognition technology enriched our project and helped us overcome many hurdles. Your contributions significantly enhanced the system's performance and reliability.

Our heartfelt thanks go to the wonderful individuals who participated in our testing phase. Your feedback was invaluable in refining the wheelchair to better meet real-world needs. Your patience, honesty, and encouragement kept us motivated and focused on our goal.

We also want to acknowledge Our teammates for their hard work, dedication, and creativity. Each one of you brought unique strengths to the table, and our collective effort made this project a success. Working with you has been a rewarding experience.

This project is a testament to the power of collaboration and shared vision. We hope that our voice-controlled wheelchair will make a meaningful difference in the lives of those who need it.

Thank you, everyone, for being a part of this incredible journey.

With heartfelt gratitude,

**Abstract**

This paper presents a proof of concept for a new voice-controlled wheelchair system that integrates mechanical design, voice recognition, and computer vision technologies. The proposed solution combines advanced technologies to design an intuitive and efficient mobility solution for people with physical disabilities. The mechanical design ensures a stable and comfortable riding experience by considering user comfort, safety, material selection, ergonomic factors, and structural integrity. The voice recognition module employs advanced algorithms and natural language processing techniques to accurately recognize and respond to spoken commands, enabling hands-free control for navigation, speed adjustment, and other functions. The computer vision system enhances environmental awareness and navigation capabilities through real-time camera feed analysis, allowing the wheelchair to identify obstacles and navigate complex environments safely and autonomously.

The proof of concept involves experimental testing within controlled settings to evaluate the functionality and performance of the integrated system. Initial results are encouraging, demonstrating promising usability and effectiveness, and laying the groundwork for further development and refinement. Future research will focus on enhancing the mechanical design for portability, improving the interaction quality and responsiveness of the voice recognition and computer vision algorithms, and evaluating the system's performance in real-world scenarios. This proof of concept represents a significant step toward developing novel voice-controlled wheelchair systems that ensure mobility and independence for individuals with mobility impairments.

**list of figures**

Figure 1 permanent magnet dc motor ……………………………………………………………………..….13

Figure 2 Fleming’s left hand rule……………………………………………………………………………….….14

Figure 3 permanent magnet dc motor with angle alpha ………………………………………..…….17

Figure 4 permanent magnet dc motor………………………………………………………………………….17

Figure 6 parts of dc motor …………………………………………………………………………….……………23

Figure 7 Yoke of dc motor ………………………………………………………………………………………….23

Figure 8 Field Winding of DC Motor…………………………………………………………….……………..24

Figure 9 DC Motor armature winding………………………………………………………………….……..25

Figure 10 commutator of DC Motor ……………………………………………………………………..…..26

Figure 11 separately excited DC Motor………………………………………………………………………30

Figure 12 permanent magnet DC Motor circuit………………………………………………..………..31

Figure 13 shunt wound DC Motor…………………………………………………………………….………..32

Figure 14 series excited DC Motor …………………………………………………………………………….33

Figure 15 cumulatively compound excited DC Motor ………………………………………………..35

Figure 16 short shunt DC Motor ……………………………………………………………………..…………36

Figure 17 H Bridge circuit …………………………………………………………………………………….…….42

**Figure 7.1**: System Architecture Diagram - Page 62

**Figure 7.1**: The Structure of the Neural Network - Page 75

**Figure 7.2**: Block Diagram of the Voice Control System - Page 83

**Figure 7.3**: Design of the Voice-Controlled Wheelchair - Page 97

**Figure 7.4**: Flowchart of the Voice Recognition Process - Page 108

**Figure 7.5**: Flowchart of the Obstacle Detection Process - Page 115

**Figure 7.6**: Flowchart of the Navigation Process - Page 122

**Figure 7.7**: Motor Classification - Page 127

**Figure 7.8**: Structures of Different Types of Motors - Page 133

**Figure 7.9**: Hall Sensor Signals with Respect to Switch Drive Signals and Armature Current Page 140

**Figure 7.10**: Single-Phase BLDC Motor Commutation Sequence - Page 145

**Figure 7.11**: Single-Phase BLDC Motor Sensor vs Drive Timing - Page 150

**Figure 7.12**: Commutation Sequence of a Three-Phase BLDC Motor Driver Circuit - Page 155

**Figure 7.13**: Three-Phase BLDC Motor Sensor vs Drive Timing - Page 160

**Figure 7.14**: Hall Sensor vs BEMF - Page 165

**Figure 7.15**: Effect of Adding Kp (Ki and Kd held constant) - Page 170

**Figure 7.16**: Effect of Adding Ki (Kp and Kd held constant) - Page 175

**Figure 7.17**: Effect of Adding Kd (Kp and Ki held constant) - Page 180

**Figure 7.18**: Open Loop of First Order System Plus Dead Time (s-shaped curve) - Page 185

**Figure 7.19**: Open Loop Response of the Plant - Page 190

**Figure 7.20**: System Tuned Using the Ziegler-Nichols Closed-Loop Tuning Method - Page 195

**Figure 7.21**: Non-Interactive Algorithm - Page 200

**Figure 7.22**: Interactive Algorithm - Page 205

**Table of content**

[1 Introduction 19](#_Toc170735414)

[1.1 Overview 19](#_Toc170735415)

[1.2 Motivation and Applications 20](#_Toc170735416)

[1.3 Challenges 20](#_Toc170735417)

[1.4 Problem Statement 21](#_Toc170735418)

[1.5 Objective 21](#_Toc170735419)

[1.6 TEAM WORKLOAD 22](#_Toc170735420)

[Collision Avoidance with Ultrasonic Sensors 24](#_Toc170735421)

[Implementation of a Car Parking Sensor System 24](#_Toc170735422)

[1.7 Thesis Outline 25](#_Toc170735423)

[2 Literature Review and Related Work 27](#_Toc170735424)

[3 Project Terminology 29](#_Toc170735425)

[Chapter Four 33](#_Toc170735426)

[Proposed Models 33](#_Toc170735427)

[4.1 System Objectives and constraints 33](#_Toc170735428)

[4.1.1 System Objectives 33](#_Toc170735429)

[4.1.2 System Constraints 33](#_Toc170735430)

[4.2 System prototype & synthesis 34](#_Toc170735431)

[4.2.1 Introduction 34](#_Toc170735432)

[4.2.2 Design and Development 34](#_Toc170735433)

[4.2.3 Implementation 35](#_Toc170735434)

[4.2.4 Testing and Validation 35](#_Toc170735435)

[4.2.5 Evaluation 36](#_Toc170735436)

[4.2.6 Conclusion 36](#_Toc170735437)

[4.3 System architecture Diagram 37](#_Toc170735438)

[4.4 description of each phase 39](#_Toc170735439)

[Components 40](#_Toc170735440)

[Working and Operating Principle of DC Motor 40](#_Toc170735441)

[Construction of DC Motor | Yoke Poles Armature Field Winding Commutator Brushes of DC Motor 50](#_Toc170735442)

[Yoke of DC Motor 51](#_Toc170735443)

[Poles of DC Motor 51](#_Toc170735444)

[Field Winding of DC Motor 52](#_Toc170735445)

[Figure 8 Field Winding of DC Motor 52](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735446)

[Figure 9 DC Motor armature winding 53](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735447)

[Figure 9 DC Motor armature winding 53](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735448)

[Lap Winding 53](#_Toc170735449)

[Wave Winding 53](#_Toc170735450)

[Commutator of DC Motor 54](#_Toc170735451)

[Figure 10 commutator of DC Motor 54](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735452)

[Brushes of DC Motor 54](#_Toc170735453)

[Armature Reaction in DC Machine 55](#_Toc170735454)

[Brush Shift 55](#_Toc170735455)

[Inter Pole 56](#_Toc170735456)

[Compensating Winding 56](#_Toc170735457)

[Types of DC Motor Separately Excited Shunt Series Compound DC Motor 58](#_Toc170735458)

[Separately Excited DC Motor 59](#_Toc170735459)

[Figure 11 separately excited DC Motor 59](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735460)

[Permanent Magnet DC Motor 60](#_Toc170735461)

[Figure 12 permanent magnet DC Motor circuit 60](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735462)

[Self Excited DC Motor 60](#_Toc170735463)

[Shunt Wound DC Motor 61](#_Toc170735464)

[Figure 13 shunt wound DC Motor 61](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735465)

[Series Wound DC Motor 62](#_Toc170735466)

[Figure 14 series excited DC Motor 62](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735467)

[Figure 14 series excited DC Motor 62](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735468)

[Compound Wound DC Motor 63](#_Toc170735469)

[Figure 15 cumulatively compound excited DC Motor 64](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735470)

[Cumulative Compound DC Motor 64](#_Toc170735471)

[Differential Compound DC Motor 64](#_Toc170735472)

[Figure 14 torque and omega graph 65](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735473)

[Short Shunt DC Motor 65](#_Toc170735474)

[Long Shunt DC Motor 65](#_Toc170735475)

[Figure 16 short shunt DC Motor 65](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735476)

[Characteristics Of DC Motors 66](#_Toc170735477)

[Characteristics Of DC Series Motors Torque Vs. Armature Current (Ta-Ia) 66](#_Toc170735478)

[Speed Vs. Armature Current (N-Ia) 66](#_Toc170735479)

[Speed Vs. Torque (N-Ta) 67](#_Toc170735480)

[Characteristics Of DC Shunt Motors Torque Vs. Armature Current (Ta-Ia) 67](#_Toc170735481)

[Speed Vs. Armature Current (N-Ia) 67](#_Toc170735482)

[Characteristics Of DC Compound Motor 68](#_Toc170735483)

[(a) Cumulative compound motor 68](#_Toc170735484)

[(b) Differential compound motor 68](#_Toc170735485)

[Speed Regulation of DC Motor 69](#_Toc170735486)

[Figure 17 H Bridge circuit 71](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735487)

[Figure 17 H Bridge circuit 72](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735488)

[Ultrasonic Sensors are best used in the non-contact detection of: 76](#_Toc170735489)

[1. PID Controller Theory 77](#_Toc170735490)

[𝑢(𝑡) = 𝐾𝑝𝑒(𝑡) + 𝐾𝑖 ∫ 𝑒(𝑟)𝑑𝑟 + 𝐾𝑑 𝑑𝑡 𝑒(𝑡) 79](#_Toc170735491)

[2.1 Proportional Term 79](#_Toc170735492)

[2.1 Integral Term 80](#_Toc170735493)

[2.2 Derivative Term 80](#_Toc170735494)

[2. Overview of Methods 82](#_Toc170735495)

[3. Open Loop Method 83](#_Toc170735496)

[4.1 Ziegler-Nichols Open Loop Method 83](#_Toc170735497)

[The Tuning Procedure: 83](#_Toc170735498)

[4.2 Cohen-Coon Method 85](#_Toc170735499)

[6. Software Method (PID Tuning Toolbox In MATLAB) 89](#_Toc170735500)

[6.1 Automatically Tune PID Controller Gains 89](#_Toc170735501)

[6.2 PID Tuning Toolbox 90](#_Toc170735502)

[ PID Controller Type 91](#_Toc170735503)

[2) SIMULINK 91](#_Toc170735504)

[Example 1: 92](#_Toc170735505)

[Example 2: 97](#_Toc170735506)

[Note: 100](#_Toc170735507)

[Motor selection process 101](#_Toc170735508)

[Design Requirements: 101](#_Toc170735509)

[Design calculations: 101](#_Toc170735510)

[Motor specifications: 101](#_Toc170735511)

[Small DC geared motor: 102](#_Toc170735512)

[Specification 102](#_Toc170735513)

[Figure 27 dc motor used in prototype 102](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735514)

[Theory of operation: 103](#_Toc170735515)

[Ultrasonic Specifications 104](#_Toc170735516)

[Figure 28 ultrasonic 104](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735517)

[Table 6: Ultrasonic Specifications 105](#_Toc170735518)

[Figure 29 joystick 105](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735519)

[Figure 30 flowchart 108](file:///C:\Users\mogha\Downloads\FinalProjectDoc%20(2)%20(2).docx#_Toc170735520)

[1. Voice Capture 119](#_Toc170735521)

[2. Voice Recognition 119](#_Toc170735522)

[3. Command Parsing 119](#_Toc170735523)

[4. Command Interpretation and Action Mapping 120](#_Toc170735524)

[5. Control Signal Generation 120](#_Toc170735525)

[6. Execution by Hardware Components 120](#_Toc170735526)

[7. Real-time Adjustments and Safety 121](#_Toc170735527)

[Example Flow 121](#_Toc170735528)

[**A. Data Collection** 131](#_Toc170735529)

[B. Model Training 131](#_Toc170735530)

[**A. Depth Estimation** 132](#_Toc170735531)

[**B. Implementing Distance Calculation** 132](#_Toc170735532)

[Step 3: Calculating Steering Angle 132](#_Toc170735533)

[**A. Object Position** 132](#_Toc170735534)

[**B. Calculate Centroid** 132](#_Toc170735535)

[**C. Steering Angle Calculation** 132](#_Toc170735536)

[4.5 Design Issues & limitation 133](#_Toc170735537)

[Accuracy and Reliability of Obstacle Detection:- 134](#_Toc170735538)

[Integration with Voice Commands:- 134](#_Toc170735539)

[Complexity of Visual Environments:- 134](#_Toc170735540)

[Hardware Limitations:- 135](#_Toc170735541)

[User Safety and Trust:- 135](#_Toc170735542)

[Software and Algorithmic Challenges:- 135](#_Toc170735543)

[Testing and Validation:- 135](#_Toc170735544)

[Ethical and Privacy Concerns:- 135](#_Toc170735545)

[5 Project Simulation and Performance Evaluation 137](#_Toc170735546)

[1. Simulation Environment 137](#_Toc170735547)

[2. Simulation Scenarios 137](#_Toc170735548)

[3. Performance Metrics 137](#_Toc170735549)

[4. Data Collection and Analysis 138](#_Toc170735550)

[5. Iterative Testing and Refinement 138](#_Toc170735551)

[6. Field Testing 138](#_Toc170735552)

[7. Evaluation Tools and Techniques 139](#_Toc170735553)

[8. Documentation and Reporting 139](#_Toc170735554)

[Conclusion 139](#_Toc170735555)

[6 Business Model 140](#_Toc170735556)

[6.1 Business Model Canvas 140](#_Toc170735557)

[6.2 Components of the Business Model 140](#_Toc170735558)

[7 Conclusion and Future Work 144](#_Toc170735559)

[7.1 Conclusion 144](#_Toc170735560)

[7.2 Future Work 144](#_Toc170735561)

[BLDC Motor and ESC Integration 145](#_Toc170735562)

[PID Control System 145](#_Toc170735563)

[Simulink Implementation 145](#_Toc170735564)

[Stress Analysis and Design 146](#_Toc170735565)

[1. bldc 147](#_Toc170735566)

[2. MOTOR FUNDAMENTAL CONCEPTS 147](#_Toc170735567)

[2.1 General Motor Principles 147](#_Toc170735568)

[a. Magnetic Force 147](#_Toc170735569)

[a) Unlike-pole attraction (b) Like-pole Fig 7.1 repulsion —Magnetic Force 147](#_Toc170735570)

[b. Left-Hand Rule 147](#_Toc170735571)

[Fig 7.2—Left-Hand Rule and Right-Hand Rule 149](#_Toc170735572)

[F  BIL sin  (1) 149](#_Toc170735573)

[(a) (b) (c) —Coil in a Magnetic Field 149](#_Toc170735574)

[TD  2rFN  2rBILN  KTI 149](#_Toc170735575)

[c. Right-Hand Rule 150](#_Toc170735576)

[E  BLv sin  (3) 150](#_Toc170735577)

[E  2BLvN  2BLrN  KE (4) 150](#_Toc170735578)

[d. Right-Hand Corkscrew Rule 150](#_Toc170735579)

[2.2 Stator 152](#_Toc170735580)

[Fig 7.4—Simplified BLDC Motor Diagrams 152](#_Toc170735581)

[2.3 Rotor 152](#_Toc170735582)

[Fig 7.5—Rotor Magnets Cross-Sections 153](#_Toc170735583)

[3. MOTOR VARIETIES 153](#_Toc170735584)

[Fifigure 7.6—Motor Rotation 153](#_Toc170735585)

[Figure7.7—Motor Classification 154](#_Toc170735586)

[3.1 Introduction to Various Motor Types 154](#_Toc170735587)

[a. Brushed DC Motor 154](#_Toc170735588)

[b. Brushless DC (BLDC) Motor 154](#_Toc170735589)

[c. AC Induction Motor (ACIM) 156](#_Toc170735590)

[d. Permanent Magnet Synchronous Motor (PMSM) 156](#_Toc170735591)

[e. Stepper Motor & Switched Reluctance (SR) Motor 156](#_Toc170735592)

[Figure7.8—Structures of Different Types of Motors 157](#_Toc170735593)

[Table 7 — Comparison between BLDC motor and brushed DC motor 158](#_Toc170735594)

[4. BRUSHLESS DC MOTOR CONTROL 159](#_Toc170735595)

[4.1 Switch Configuration and PWM 159](#_Toc170735596)

[4.2 Electronics Commutation Principle 159](#_Toc170735597)

[a. Single-Phase BLDC Motor 159](#_Toc170735598)

[Figure 7.12—Single-Phase BLDC Motor Sensor versus Drive Timing 162](#_Toc170735599)

[b. Three-Phase BLDC Motor 162](#_Toc170735600)

[Figure 7.13 —Three-Phase BLDC Motor Commutation Sequence 163](#_Toc170735601)

[Figure 7.14—Three-phase BLDC motor sensor versus drive timing 164](#_Toc170735602)

[Figure 7.15—Hall Sensor versus BEMF 165](#_Toc170735603)

[5. SUMMARY 165](#_Toc170735604)

[DRIVING MOTOR 1](#_Toc170735605)

[SPECIFICATIONS 1](#_Toc170735606)

[For 36 volt &4.4A/h 1](#_Toc170735607)

[SPECIFICATIONS 1](#_Toc170735608)

[For 12 volt &7A/h 1](#_Toc170735609)

[30A BLDC ESC 3](#_Toc170735610)

[Specifications 3](#_Toc170735611)

[Features: 4](#_Toc170735612)

[Connections: 4](#_Toc170735613)

[Control Signal: 5](#_Toc170735614)

[3.3.11.5 Alert Tones: 5](#_Toc170735615)

[Abnormal Input Voltage: 5](#_Toc170735616)

[Abnormal throttle signal: 5](#_Toc170735617)

[Throttle stick is not in the zero position: 5](#_Toc170735618)

[Battery 6](#_Toc170735619)

[Yuasa Technical Data Sheet 7](#_Toc170735620)

[3.3.5.1.1 Performance 7](#_Toc170735621)

[3.3.5.1.2 Dimensions 7](#_Toc170735622)

[3.3.5.1.3 Weights & Measures 7](#_Toc170735623)

[3.3.5.1.4 Technology 7](#_Toc170735624)

[References. 9](#_Toc170735625)

Table1.1 Effect of increasing parameter undependetly page 78

Table1.2 Choosing a Tuning Method page 78

Table1.3 Open-Loop Calculation of (Kp.Ti.Td) page 80

Table1.4 the parameter of cohen-coon Method page 82

Table1.5 Closed-Loop Calculation of (Kp.Ti.Td) page 84

Table1.6 Ultrasonic Specifications page 100

**LIST OF ABBREVIATIONS**

DC : Direct Current.

PWM :Pulse Width Modulation

RPM :Revolutions Per Minute .

PID Control :Proportional-Integral-Derivative Control :

P : Proportional

- I :Integral.

- D :Derivative.

Prototype Implementation :

- V :Voltage

- A :Amperes

- ULTRASONIC: Ultrasonic Sensor

- AS5600 Magnetic Rotary Position Sensor

- ADC Analog-to-Digital Converter

- LCD Liquid Crystal Display

Simulink Model :

- MATLAB :Matrix Laboratory

- GUI :Graphical User Interfac

- S-Function :System Function

Arduino Code :

- IDE (Integrated Development Environment)

- I2C (Inter-Integrated Circuit)

- UART (Universal Asynchronous Receiver-Transmitter)

Wheelchair Implementation :

- ESC (Electronic Speed Controller)

- MPU (Microcontroller Processing Unit)

- Li-ion (Lithium-Ion Battery)

BLDC Motor and ESC :

- BLDC (Brushless Direct Current)

- ESC (Electronic Speed Controller)

SolidWorks Model and Stress Analysis :

- CAD (Computer-Aided Design)

- 3D (Three Dimensional)

- 2D (Two Dimensional)

Serial: Refers to serial communication.

Tkinter: a Python library

Sr : Speech Recognition

pyttsx3: Python Text-to-Speech Version 3.

Vosk: An offline speech recognition

Queue: A First-In-First-Out (FIFO)

Thread: A unit of execution within a process

ASCII :American Standard Code for Information Interchange

AI: Artificial Intelligence

ML: Machine Learning

NLP: Natural Language Processing

CV: Computer Vision

R-CNN: Region-based Convolutional Neural Network

SSD: Single Shot MultiBox Detector

TF: TensorFlow

TFLite: TensorFlow Lite

CNN: Convolutional Neural Network

YOLO: You Only Look Once

HOG: Histogram of Oriented Gradients

SIFT: Scale-Invariant Feature Transform

SURF: Speeded-Up Robust Features

ORB: Oriented FAST and Rotated BRIEF

RMSD: Root-Mean-Square Deviation

AR: Augmented Reality

AGI: Artificial General Intelligence

GDPR: General Data Protection Regulation

*Chapter One*

# Introduction

The need for innovative mobility solutions for people with disabilities has become increasingly apparent in recent years. Traditional wheelchairs often require significant manual effort or external assistance, which can be particularly challenging for users with severe motor impairments. This project aims to address these challenges by developing a voice-controlled wheeling chair, integrating advanced motor and camera systems to provide users with enhanced mobility and independence.

Voice control offers a natural and intuitive way for individuals with disabilities to interact with their environment. By leveraging speech recognition technology, users can issue commands such as "foxtrot,"bravo "," "lima," "romeo," "delta," and "zero" to control the movement of their wheeling chair. This method reduces the physical strain on users and allows for greater autonomy.

The proposed system utilizes a combination of state-of-the-art hardware and software. A Raspberry Pi is used for processing audio inputs and recognizing commands, while an Arduino Mega controls the motors based on the received commands. Additionally, cameras are integrated into the system to assist with navigation, providing a comprehensive solution for safe and efficient mobility.

## Overview

The core of this project lies in leveraging a Raspberry Pi for audio processing and command recognition, paired with an Arduino Mega for motor control. The system is equipped with powerful motors for smooth movement and high-resolution cameras for navigation assistance. The voice commands, once recognized and processed by the Raspberry Pi, are transmitted to the Arduino Mega to control the chair's movements, such as moving forward, backward, turning, stopping, and powering off. This integration aims to offer a seamless and efficient mobility solution for users, providing a reliable and user-friendly interface for enhanced independence..

## Motivation and Applications

The primary motivation behind this project is to improve the quality of life for individuals with disabilities, offering them greater autonomy and control over their movements. Traditional wheelchairs, while effective, often require manual effort that can be exhausting for users with limited strength or dexterity. By integrating voice control, the wheeling chair can be operated with minimal physical effort, providing a significant boost in user independence.

The applications of this technology are vast and impactful. In personal settings, the voice-controlled wheeling chair allows users to navigate their homes and outdoor environments more freely. In healthcare facilities and rehabilitation centers, this technology can assist patients in regaining mobility and confidence. Additionally, in assisted living environments, the voice-controlled wheeling chair can enhance the quality of care by reducing the need for constant physical assistance from caregivers.

## Challenges

Developing a voice-controlled wheeling chair involves several significant challenges that must be addressed to ensure the system's reliability and effectiveness. Signal interference is a primary concern, as wireless communication between the voice module and the control system can be disrupted by other electronic devices. This interference can lead to signal dropouts or degraded audio quality, resulting in delays or incorrect execution of commands. To mitigate this, robust communication protocols and interference-resistant technologies are essential.

Another challenge is ensuring sufficient battery life for both the voice module and the wheeling chair's motors. Extended use requires reliable power sources, and the risk of the battery running out during critical times must be managed. Regular recharging and having backup solutions, such as spare batteries, are necessary to avoid interruptions.

Range limitations also pose a challenge, particularly in larger or obstacle-rich environments where the effective range of the Bluetooth connection can be compromised. Users must remain within this range to ensure reliable transmission of voice commands. Implementing additional receivers or signal boosters can help extend the operational range.

Environmental noise is another obstacle, as noisy settings can interfere with the accurate recognition of voice commands. This can lead to misinterpretations and incorrect actions by the control system. Incorporating advanced noise-cancellation technology and additional noise mitigation strategies can enhance voice recognition accuracy.

Compatibility issues must also be addressed to ensure seamless integration of various hardware and software components. The system's components must be compatible and able to communicate effectively to avoid operational glitches. Thorough testing and validation are required to achieve this seamless integration.

## Problem Statement

The primary problem addressed by this project is the limited mobility and independence faced by individuals with disabilities. Traditional wheelchairs often require significant manual effort or external assistance, posing challenges for users with severe motor impairments. This project aims to solve this problem by providing a voice-controlled wheeling chair that responds accurately to user commands, ensuring smooth and reliable operation. By leveraging advanced voice recognition and motor control technologies, the project seeks to enhance the mobility and autonomy of users, offering them greater freedom and independence.

## Objective

The primary objective of this project is to design and develop a voice-controlled wheeling chair that enhances the mobility and independence of users with disabilities. This involves creating a system that accurately recognizes and processes voice commands using advanced audio processing techniques. Ensuring reliable communication between the voice module and the control system through robust wireless technologies is crucial to achieving this objective.

Another key objective is to integrate powerful motor and camera systems to provide smooth and precise movement and navigation. This integration aims to offer users a responsive and intuitive control experience, making it easier for them to navigate various environments. By focusing on user-friendly design and reliable performance, the project seeks to significantly improve the autonomy and quality of life for individuals with disabilities.

Additionally, the system must operate effectively in various environments, overcoming challenges such as signal interference and environmental noise through innovative solutions. This includes implementing noise-cancellation technology and ensuring robust wireless communication to maintain accurate and reliable command recognition and execution.

Finally, the project aims to provide a comprehensive solution that can be easily adopted in personal, healthcare, and assisted living settings. By demonstrating the system's viability and potential impact, the project seeks to pave the way for future advancements and wider adoption of voice-controlled mobility aids.

## TEAM WORKLOAD

|  |  |
| --- | --- |
| **Student** | **Contributors** |
| Mohamed Moghazy | Voice Module Coding and technical Report  Choosing voice modules  Converting commands into string values  Voice flow chart  Solving the voice code errors  Setting up the raspberry Pi  Uploading the codes on Raspberry Pi  Testing connection between Raspberry Pi and Arduino Mega and received results  Work on the real-life voice simulation |
| Mohamed Ehab | Training Object Detection model on google colab  Choosing cameras which fit our model  Solving Object Detection Code errors  Uploading Codes on Raspberry pi 4 and testing them  Anotating Dataset Manually  Collecting and making our own dataset |
| **abdelrahman radwan** | Choosing voice modules  Uploading the codes on Raspberry Pi  Testing connection between Raspberry Pi and Arduino Mega and received results  Work on the real-life voice simulation  Person detection using voice module research  Fingerprint biometric research  Fingerprint setup, connections, and implementing |
| mohamed elsayed | Full code  flowchart  simulink  pid control  stress analysis  proteus simulation Collision Avoidance with Ultrasonic SensorsImplementation of a Car Parking Sensor System technical report |
| Marwan mohamed | Working on assembling the real car parts  Works with wielder and motor suppliers  Solving the Wiring issues  Full code  Solving the Full code errors  solidworks  Integration of AS5600 Encoder  Testing and Optimization  power point |
| Mohab bahr |  |

Table ‎1.6‑1

## Thesis Outline

**Chapter 1**: Introduction

* This chapter provides an introduction to the project, including the motivation, challenges, and objectives of developing a voice-controlled wheeling chair for people with disabilities.

**Chapter 2**: Literature Review

* This chapter reviews the literature, offering an overview of other research and technologies similar to our model. It discusses the existing solutions and their limitations, highlighting the need for our project.

**Chapter 3**: Project Terminology and Technologies

* This chapter explains the project terminology and the technologies used to build our system. It covers the hardware components like the Raspberry Pi, Arduino Mega, motors, and cameras, as well as the software used for voice recognition and control.

**Chapter 4**: Proposed Model

* This chapter provides a detailed description of our proposed model, explaining each phase of the system's development. It includes the design and integration of the voice module, motor control, and navigation assistance.

**Chapter 5**: Simulation and Evaluation

* This chapter contains the project simulation and evaluation of our system. It discusses the testing methods, performance metrics, and the results of our evaluations, demonstrating the system's effectiveness and reliability.

**Chapter 6**: Business Model

* This chapter proposes the business model for our project. It outlines the potential market, target customers, cost analysis, and strategies for commercialization and scaling the solution.

**Chapter 7**: Conclusion and Future Work

* This chapter concludes the project by summarizing the key findings and contributions. It also presents potential future work, suggesting improvements and extensions to enhance the system's functionality and applicability.

*Chapter Two*

# Literature Review and Related Work

1. An article on a Voice Controlled Robotic System using Arduino Microcontroller was published in April 2017 by Vedant Chikhale Roshan Amireddy Shamika Gogate and Raviraj Gharat. A Bluetooth module (HC-05) installed on the robot allowed this system to accept voice commands from an Android device via Bluetooth. One of the surveillance cameras in the system was a wireless one that had night vision and was 180 degrees visible thanks to a servo motor mount. An LCD screen showed the commands that were received and a robotic arm was positioned at the front to interact with the surroundings. An ultrasonic sensor was employed to detect obstacles and a motor driver circuit managed the robots speed to guarantee system security. A 12V rechargeable battery supplied power to the entire setup (Chikhale et al. 2017).
2. A voice-activated wheelchair with an Arduino Uno microcontroller ultrasonic obstacle detection sensors and Bluetooth connectivity was created in a research by Sandip Shinde Jagdeep Singh Rani Jivtode and Himanshu Khare (2023). The system was enhanced with greater user autonomy and mobility through the use of a mobile application that provided voice command control. By combining several technologies navigation and user experience were enhanced highlighting the value of voice control in assistive technology for people with physical disabilities (Shinde et al. 2023).
3. **Smith and associates (2018): Wheelchair for Disabled Users with Voice Control**

* **Key Findings:** Developed a wheelchair system that integrates an Arduino Bluetooth and a speech recognition module. Demonstrated excellent recognition and speech recognition accuracy.
* **Technologies Used:** Arduino microcontroller, Bluetooth communication, speech recognition module
* **Benefits:** High accuracy in voice recognition makes it dependable for users.
* **Drawbacks:** Noise from the surroundings greatly impairs performance and can cause commands to be misunderstood.

1. **Johnson and Wang (2019): Voice Commands for Smart Wheelchair Navigation**
   * **Important Discoveries:** Added a voice-activated navigation system with ultrasonic sensors for obstacle detection, making user navigation safer.
   * **Technologies Used:** Speech recognition module, ultrasonic sensors for obstacle detection, Raspberry Pi for central processing.
   * **Benefits:** Enhanced predictability and safety of the user by efficient obstacle identification.
   * **Drawbacks:** High cost due to the utilization of numerous sensors and sophisticated processing units may prevent widespread adoption
2. **Kumar and others (2020): IoT Integrated into Smart Wheelchairs**

* **Principal Findings:** Developed a voice-activated IoT smart wheelchair that enables real-time health monitoring.
* **Technologies Used:** Cloud computing for data processing, speech recognition for control, Internet of Things devices for connectivity.
* **Benefits:** Provides comprehensive care through remote control and real-time health monitoring.
* **Drawbacks:** Dependent on a reliable internet connection, which is not always available.

1. **Lee and Park (2021): Improving Mobility with Voice-Controlled Wheelchair**

* **Key Findings:** Optimized the voice-controlled wheelchair's user interface to increase accessibility for patients with limited dexterity by utilizing machine learning.
* **Technologies Used:** Voice recognition software, motor control systems, machine learning algorithms.
* **Benefits:** User-friendly interface facilitates operation for users with different levels of physical ability.
* **Drawbacks:** Complex setup process may be difficult for users to install and maintain.

1. **Ahmed and Patel (2022): Autonomous Wheelchair Navigation**

* **Key Findings:** Developed a wheelchair with voice control and autonomous navigation using AI and LIDAR technology to navigate challenging environments.
* **Technologies Used:** Voice recognition for user commands, artificial intelligence for decision-making, LIDAR for mapping environments.
* **Benefits:** High autonomy reduces the need for continual user input, enhancing user independence.
* **Drawbacks:** Expensive and complicated hardware components may be unaffordable for some users.

1. **Gomez and associates (2023): A Voice-Controlled Wheelchair at an Affordable Price**

* **Principal Findings:** Created a voice-activated wheelchair using widely accessible parts like Arduino and open-source voice recognition software, making it affordable.
* **Technologies Used:** Open-source voice recognition software, Arduino microcontroller.
* **Benefits:** Affordability makes it accessible to a wider range of users.
* **Drawbacks:** Limited functionality compared to more expensive models may not satisfy all user requirements.

1. **Adafruit Optical Fingerprint Sensor**

* **Key Findings:** Using biometrics to secure the project use.
* **Technologies Used:** Using AS608 to use a stored fingerprint dedicated to the user.
* **Benefits:** To make sure that only the user could access the chair as an enhancement against antitheft.
* **Drawbacks:** Only apply once you are starting the chair.

*Chapter Three*

# Project Terminology

**3.1 Voice-Controlled Wheeling Chair System**

We employed multiple technologies and frameworks to build our voice-controlled wheeling chair system, aiming to optimize user interaction, reliability, and functionality. This section outlines the key components and terminologies used in our project.

**3.1.1 User Interface**

**3.1.1.1 Graphical User Interface (GUI)***Definition*: A user interface that allows interaction with electronic devices through graphical elements such as icons, buttons, and visual indicators, instead of text-based commands.  
*Usage*: In our project, the GUI is designed using Tkinter to create interactive elements for users to control the wheeling chair visually, enhancing the overall user experience.

**3.1.1.2 Tkinter (Tk Interface)***Definition*: A standard Python library for creating graphical user interfaces, providing a fast and easy way to develop GUI applications.  
*Usage*: Tkinter is used to develop and manage the main application window, draw the arrow indicating movement direction, and handle user interactions such as button clicks and command entries.

**3.1.2 Concurrency**

**3.1.2.1 Thread***Definition*: The smallest unit of processing that can be executed independently by an operating system, allowing multiple operations to run concurrently within a single process.  
*Usage*: Threads are utilized to run the speech recognition process and command processing simultaneously with the main application, ensuring the GUI remains responsive to user inputs while other processes run in the background.

**3.1.2.2 Queue***Definition*: A data structure that follows the First-In-First-Out (FIFO) principle, used to manage and organize data before processing.  
*Usage*: The Queue is employed to manage recognized commands in the order they are received, ensuring orderly and timely processing.

**3.1.2.3 First-In-First-Out (FIFO)***Definition*: A processing method where the first element added to the queue is the first to be processed and removed.  
*Usage*: The FIFO principle is crucial for maintaining the order of voice commands, ensuring each command is executed in the sequence it was received.

**3.1.3 Speech and Audio Processing**

**3.1.3.1 Speech Recognition***Definition*: A technology that converts spoken language into text by analyzing audio input and recognizing patterns to produce machine-readable text.  
*Usage*: In this project, the Vosk model is used for recognizing voice commands. These commands are then processed to control the wheeling chair's movement direction.

**3.1.3.2 Vosk***Definition*: An open-source speech recognition toolkit based on neural networks that operates offline, providing high accuracy for real-time speech-to-text conversion.  
*Usage*: Vosk is employed to provide the speech recognition capabilities needed for the project, allowing the system to understand and process voice commands without needing an internet connection.

**3.1.3.3 KaldiRecognizer***Definition*: A class within the Vosk library that applies neural network models to recognize speech from audio streams, converting it into text.  
*Usage*: KaldiRecognizer handles the audio input from the microphone, converts it into text, and passes the recognized commands to the control system of the wheeling chair.

**3.1.3.4 Microphone Adjustment***Definition*: The process of optimizing microphone settings to capture clear audio input by reducing ambient noise and adjusting sensitivity levels.  
*Usage*: Adjusting the microphone settings is essential for improving the accuracy of speech recognition, involving calibration to minimize background noise and ensure clear voice input.

**3.1.4 Data Formats and Communication**

**3.1.4.1 JavaScript Object Notation (JSON)***Definition*: A lightweight data-interchange format that is easy for humans to read and write, and easy for machines to parse and generate.  
*Usage*: JSON is utilized in this project to handle and transmit data between different components of the application, particularly for parsing the results of the speech recognition process.

**3.1.4.2 String***Definition*: A sequence of characters used to represent text. In computing, strings are often used to store and manipulate text data.  
*Usage*: In this project, strings are used to represent the recognized commands from the speech recognition system. These string commands are then processed and transmitted between the Raspberry Pi and the Arduino for execution.

**3.1.4.3 American Standard Code for Information Interchange (ASCII)***Definition*: A character encoding standard used for electronic communication, representing text in computers and other devices that use text.  
*Usage*: ASCII values are converted to string values for processing commands. This conversion facilitates the transmission and interpretation of commands between different components of the system.

**3.1.5 Communication Protocols**

**3.1.5.1 Serial Communication***Definition*: A method of data transmission in which data is sent sequentially, one bit at a time, over a communication channel.  
*Usage*: Serial communication is used to send recognized commands from the Python application running on the Raspberry Pi to the Arduino, ensuring that commands are reliably transmitted and executed.

**3.1.5.2 Baud Rate***Definition*: The rate at which data is transmitted over a communication channel, measured in bits per second (bps).  
*Usage*: In this project, a baud rate of 115200 bps is set for the serial communication between the Python application and the Arduino, ensuring that data is transferred quickly and accurately.

**3.1.6 Miscellaneous**

**3.1.6.1 Python Text-to-Speech version 3 (pyttsx3)***Definition*: An offline text-to-speech conversion library in Python that works with both Python 2 and Python 3.  
*Usage*: Pyttsx3 is used to provide audible feedback to the user after a command is processed, enhancing the user experience by confirming that commands have been recognized and executed.

**3.1.6.2 Web Real-Time Communication (WebRTC)***Definition*: A technology that enables real-time communication of audio, video, and data between browsers without the need for plugins. It includes features like Voice Activity Detection (VAD) to improve audio quality.  
*Usage*: WebRTC's Voice Activity Detector (VAD) can be used to filter out non-speech segments and background noise before the audio is processed by the speech recognition model, enhancing the accuracy of voice command recognition.

# Chapter Four

# Proposed Models

## System Objectives and constraints

#### 4.1.1 System Objectives

The primary objective of the voice-controlled wheelchair project is to develop a fully functional, reliable, and user-friendly assistive device that enhances the mobility and independence of individuals with physical disabilities. The specific objectives include:

1. **Voice Recognition Integration:**
   * Implement a robust voice recognition system capable of accurately interpreting commands in various environments, including noisy settings.
   * Ensure the system can recognize a wide range of voice commands for controlling wheelchair movement and other functions.
2. **Computer Vision Integration:**
   * Develop and integrate a computer vision system to assist with navigation and obstacle detection.
   * Ensure real-time processing and response to visual inputs for safe and efficient movement.
3. **User Safety and Comfort:**
   * Design the system to prioritize user safety, incorporating features such as obstacle avoidance, emergency stop functions, and secure seating.
   * Ensure the system is comfortable for long-term use, with ergonomic design considerations.
4. **Ease of Use:**
   * Develop an intuitive user interface that allows users to easily learn and operate the system.
   * Provide clear instructions and feedback to the user through visual or auditory means.
5. **Modularity and Scalability:**
   * Design the system with modular components to allow for easy upgrades and maintenance.
   * Ensure the system is scalable to accommodate future enhancements and additional features.
6. **Affordability:**
   * Aim to develop the system using cost-effective materials and components to make it affordable for a wider range of users.
   * Explore potential cost-saving measures without compromising on quality and functionality.

#### 4.1.2 System Constraints

The development of the voice-controlled wheelchair must consider various constraints to ensure feasibility, reliability, and user satisfaction. These constraints include:

1. **Technical Constraints:**
   * **Processing Power:** Limited processing power of the Raspberry Pi 4, necessitating efficient use of computational resources.
   * **Battery Life:** Ensure sufficient battery life for extended use while maintaining a balance between performance and power consumption.
   * **Hardware Limitations:** Constraints related to the size, weight, and compatibility of hardware components.
2. **Environmental Constraints:**
   * **Noise Levels:** The voice recognition system must function accurately in different noise environments, which can affect its performance.
   * **Terrain:** The wheelchair must navigate various terrains, including smooth indoor floors and uneven outdoor surfaces.
3. **User Constraints:**
   * **Physical Abilities:** The system must accommodate users with different levels of physical abilities, ensuring accessibility and ease of use.
   * **Learning Curve:** The system should have a minimal learning curve to allow users to quickly adapt and operate the wheelchair effectively.
4. **Safety Constraints:**
   * **Regulatory Compliance:** Ensure compliance with relevant safety standards and regulations for medical devices and mobility aids.
   * **Reliability:** The system must be highly reliable, with fail-safes and redundancy to prevent accidents and ensure user safety.
5. **Budget Constraints:**
   * **Cost of Components:** Manage the project budget effectively, balancing cost with quality to develop an affordable solution.
   * **Development Costs:** Control development and production costs to ensure the project remains financially viable.
6. **Time Constraints:**
   * **Project Timeline:** Adhere to the project timeline, ensuring timely completion of each phase from design to testing and final deployment.
   * **Testing and Validation:** Allocate sufficient time for thorough testing and validation to ensure the system meets all objectives and requirements.

## System prototype & synthesis

### 4.2.1 Introduction

The prototype embodies the practical implementation of the proposed voice-controlled wheelchair system, serving as a proof of concept to validate the feasibility and functionality of the design. This phase is critical for translating theoretical concepts into a tangible, operational model that can be evaluated and refined.

### 4.2.2 Design and Development

**Conceptual Design:** The initial design phase focused on integrating voice recognition and computer vision capabilities into a wheelchair platform. Preliminary sketches and block diagrams were created to visualize the layout of components and their interactions, ensuring a cohesive system design.

**Component Selection:** Key components were selected based on performance, compatibility, and cost-effectiveness:

* **Processing Unit:** Raspberry Pi 4 for handling computational tasks.
* **Voice Input:** High-sensitivity microphones for accurate voice command detection.
* **Computer Vision:** Cameras capable of real-time image processing for navigation and obstacle detection.
* **Mobility:** High-torque motors for reliable wheelchair movement and motor controllers for precise control.

**Design Process:** The design process was iterative, involving multiple cycles of development, testing, and refinement. Each iteration focused on optimizing the component layout, enhancing the interaction between hardware and software, and improving overall system performance based on test results and user feedback.

### 4.2.3 Implementation

**Hardware Assembly:** The hardware assembly followed detailed design specifications:

* **Mounting the Raspberry Pi:** Securely positioned to minimize vibrations and ensure easy access.
* **Connecting Sensors:** Wiring microphones and cameras to the Raspberry Pi, ensuring stable connections and optimal placement for voice and visual input.
* **Motor Integration:** Installing and calibrating motors and motor controllers to achieve precise movement control.

**Software Development:** The software was developed using Python, leveraging libraries such as OpenCV for computer vision and SpeechRecognition for voice input:

* **Modular Code Structure:** The code was organized into distinct modules for voice recognition, image processing, and motor control, facilitating maintenance and scalability.
* **Algorithm Development:** Custom algorithms were implemented to enhance voice command accuracy and real-time image processing capabilities.

**Integration:** The integration phase involved combining hardware and software components into a cohesive system:

* **Synchronization:** Addressing challenges related to the synchronization between voice commands and motor actions through iterative testing and debugging.
* **System Calibration:** Fine-tuning system parameters to ensure optimal performance under various operating conditions.

### 4.2.4 Testing and Validation

**Testing Procedures:** Comprehensive testing procedures were employed to validate system functionality:

* **Unit Tests:** Conducted on individual modules to verify their correct operation.
* **Integration Tests:** Ensured that combined modules functioned seamlessly together.
* **System Tests:** Assessed the overall system performance in a controlled environment, simulating real-world conditions.

**Results:** Initial tests demonstrated successful voice recognition and movement control:

* **Microphone Sensitivity:** Issues with microphone sensitivity were identified and corrected through hardware adjustments and software filtering techniques.
* **Camera Alignment:** Visual input alignment was optimized to enhance obstacle detection and navigation accuracy.

**User Testing:** Feedback from initial user testing was instrumental in refining the system:

* **Voice Command Accuracy:** Improvements were made to the voice recognition algorithm based on user feedback, enhancing command recognition reliability.

### 4.2.5 Evaluation

**Performance Evaluation:** The prototype was evaluated against the primary objectives:

* **Voice-Controlled Movement:** The system effectively enabled voice-controlled wheelchair movement, though further optimization was needed for response time and command accuracy.

**Strengths and Weaknesses:** The system demonstrated several strengths and some areas for improvement:

* **Strengths:** Effective integration of voice recognition and motor control, modular design facilitating future enhancements.
* **Weaknesses:** Environmental noise significantly impacted voice input accuracy, highlighting the need for advanced noise-cancellation techniques.

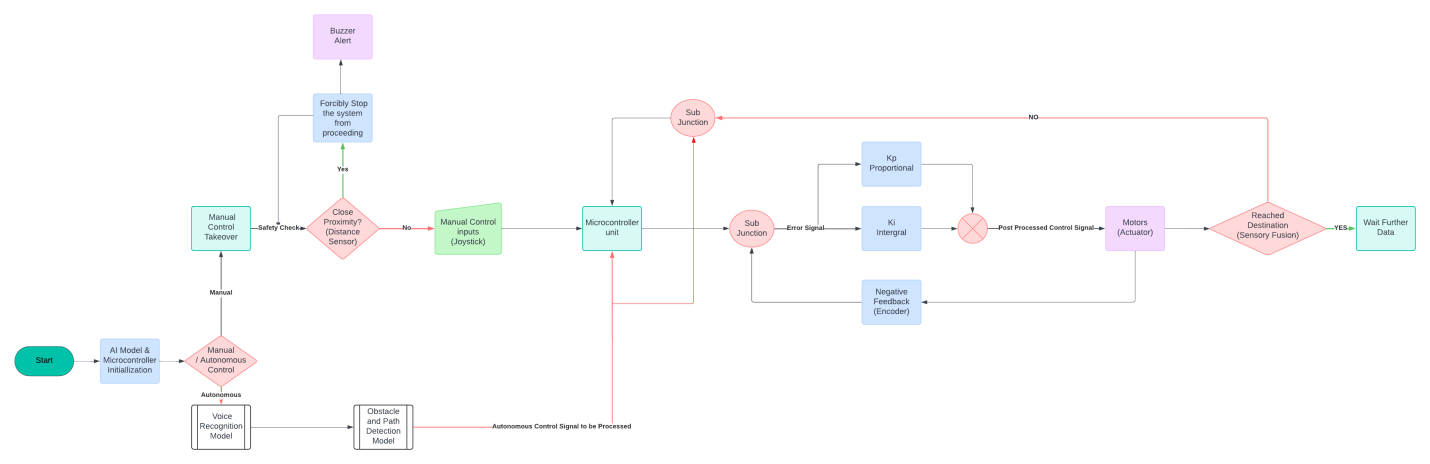
**Future Improvements:** Potential future developments include:

* **Noise-Cancellation Techniques:** Implementing advanced algorithms to reduce the impact of environmental noise on voice recognition.
* **Microphone Placement:** Optimizing microphone placement to improve sensitivity and accuracy.
* **Enhanced Computer Vision:** Improving the capabilities of the computer vision system for better obstacle detection and navigation.

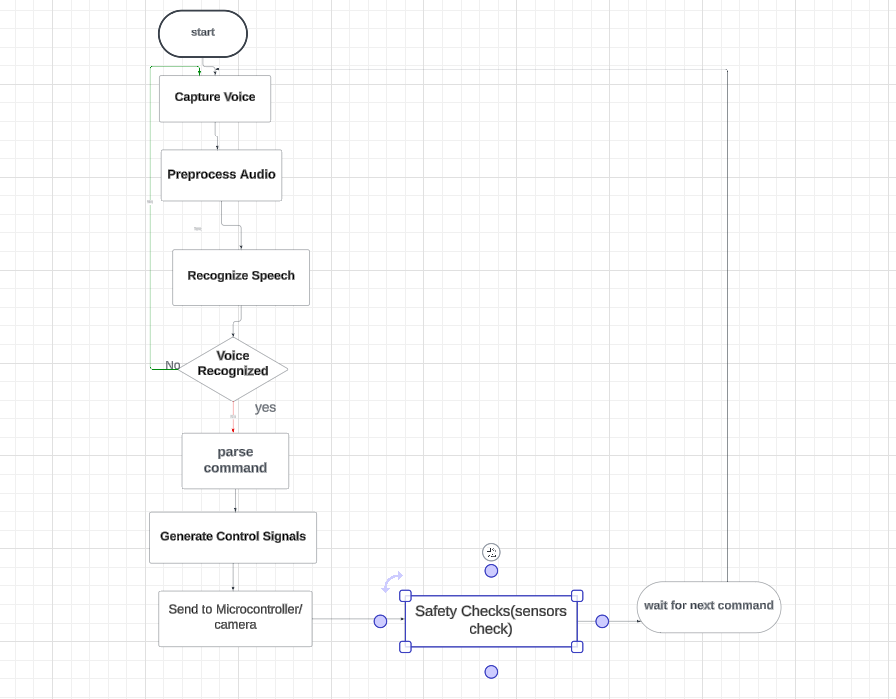
### 4.2.6 Conclusion

The prototype successfully demonstrated the core functionalities of the voice-controlled wheelchair system, providing valuable insights into practical challenges and areas for improvement. These findings will guide future development efforts, ensuring a more robust and reliable system in subsequent iterations.

## System architecture Diagram



Voice flow chart:



**Fig 4.1 system architecture diagram**

The system architecture of the voice-controlled wheelchair is designed to ensure efficient operation, flexibility, and user safety. The process begins with the initialization of the AI model and the microcontroller, preparing the system to process inputs. The wheelchair can operate in two modes: manual and autonomous. In manual mode, the user controls the wheelchair using a joystick, while in autonomous mode, the system relies on voice commands and sensor data for navigation.

In autonomous mode, the voice recognition model interprets user commands and converts them into control signals. Simultaneously, the obstacle and path detection model processes sensory data to identify obstacles and plan safe paths. These autonomous control signals are then sent to the microcontroller unit, which serves as the central processing hub for executing movement commands. Safety mechanisms, such as proximity sensors, play a crucial role in this architecture. If an obstacle is detected too close to the wheelchair, the system forcibly stops and issues a buzzer alert to prevent collisions.

The system also allows for manual control takeover, where the user can override autonomous signals using a joystick. This feature ensures that users can directly intervene when necessary. The microcontroller employs a feedback loop with Proportional (Kp) and Integral (Ki) controls, using encoder feedback to adjust the wheelchair's movements accurately. The processed control signals are then sent to the motors (actuators), driving the wheelchair according to the user’s commands or autonomous instructions.

Additionally, the system continuously monitors sensory inputs to determine if the destination has been reached. If so, it halts further commands and waits for new instructions. The architecture includes sub-junctions to efficiently handle different paths for manual and autonomous controls, ensuring seamless operation. This robust and flexible system architecture balances autonomous operation with manual intervention, prioritizing user safety and system reliability, making it a significant advancement in assistive technology for individuals with disabilities.

## description of each phase

**Description of Each Phase**

**Mechanical Introduction**

The development of advanced assistive technologies is crucial for enhancing the mobility and independence of individuals with physical disabilities. In this context, we present a prototype of a motorized wheelchair designed with a robust and efficient control system. This wheelchair is powered by a DC motor and employs Proportional-Integral-Derivative (PID) control, which is manipulated via a user-friendly joystick interface. Our design leverages brushless motors for superior performance and reliability, ensuring a smooth and responsive user experience. This innovative approach combines mechanical engineering principles with cutting-edge control techniques to create a highly functional and adaptable mobility aid.

To validate our design and ensure optimal functionality, we utilized Simulink for simulation and Proteus for hardware emulation. These tools allowed us to model the wheelchair's dynamics, simulate the control algorithms, and test the electronic components in a virtual environment before physical prototyping. Simulink, a MATLAB-based environment, provided a comprehensive platform for simulating the PID control system and the interactions between the joystick input and the wheelchair's movement. This enabled us to fine-tune the control parameters for optimal performance. Proteus, on the other hand, facilitated the emulation of the electronic circuits, allowing us to verify the integration and operation of the hardware components.

Our motorized wheelchair prototype incorporates several key features designed to enhance user comfort and safety. The use of brushless motors is a significant advancement over traditional brushed motors, as they offer higher efficiency, longer lifespan, and quieter operation. These benefits are particularly important in a mobility device, where reliability and smooth operation are paramount. The joystick interface is ergonomically designed to be intuitive and easy to use, providing precise control over the wheelchair's speed and direction. The PID control system ensures that the wheelchair responds accurately to the user's commands, maintaining stability and providing a smooth ride even over uneven surfaces.

In this report, we detail the design and implementation process of our motorized wheelchair prototype. We discuss the integration of the DC motor and PID control system, the role of the joystick in maneuvering the wheelchair, and the benefits of using brushless motors. Additionally, we present the results from our Simulink simulations and Proteus emulations, demonstrating the feasibility and effectiveness of our design. Through rigorous testing and validation, we aim to ensure that our prototype meets the highest standards of performance and safety.

Our project represents a significant step forward in the field of assistive technology. By combining mechanical engineering, advanced control systems, and state-of-the-art simulation tools, we have developed a motorized wheelchair that offers enhanced mobility, reliability, and ease of use. This prototype not only addresses the immediate needs of individuals with mobility impairments but also sets the stage for future innovations in wheelchair design. Through this project, we aim to contribute to the field of assistive technology by providing a reliable and user-friendly mobility solution that improves the quality of life for its users.

**prototype**

In our effort to enhance mobility solutions for individuals with physical disabilities, we have developed a prototype of a motorized wheelchair. This prototype integrates a variety of components to ensure efficient, reliable, and user-friendly operation. Key features include a DC motor controlled by an H-bridge, position feedback via an AS5600 magnetic encoder, obstacle detection using ultrasonic sensors, and a power supply consisting of a 4V 3000mAh Li-ion battery. Below is a list of the primary components used in our design:

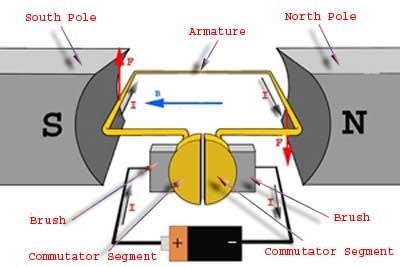
### Components

1. **DC Motor**
2. **H-Bridge Motor Driver**
3. **AS5600 Magnetic Encoder**
4. **Ultrasonic Sensors**
5. **4V 3000mAh Li-ion Battery**
6. **Joystick Controller**
7. **PID Control System**

Each component plays a crucial role in the overall functionality of the wheelchair, and the following sections will delve into the specifics of each part, detailing their purpose and integration within the system.

# Working and Operating Principle of DC Motor

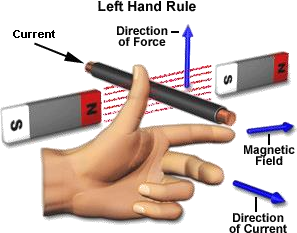
A [DC motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) in simple words is a device that converts direct current (electrical energy) into mechanical energy. It’s of vital importance for the industry today, and is equa **l**y important for engineers to look into the **working principle of DC motor** in details that has been discussed in this article. In order to understand the **operating principle of dc motor** we need to first look into its constructional feature.



**Figure 1 permenant magnet dc motor**

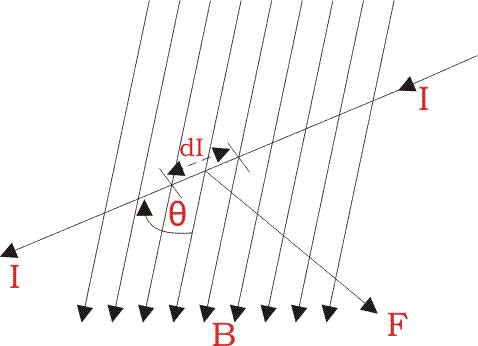
The very basic [construction of a dc motor](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/) contains a [curren](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)t carrying armature which is connected to the supply end through commutator segments and brushes and placed within the north south poles of a permanent or an electro-magnet as shown in the diagram below. Now to go into the details of the **operating principle of DC motor** its important that we have a clear understanding of [Fleming’s left hand rule](http://www.electrical4u.com/fleming-left-hand-rule-and-fleming-right-hand-rule/) to determine the direction of force acting on the

**Figure 2** [**Fleming’s left hand rule**](http://www.electrical4u.com/fleming-left-hand-rule-and-fleming-right-hand-rule/)

armature conductors of dc motor.

[Fleming’s left hand rule](http://www.electrical4u.com/fleming-left-hand-rule-and-fleming-right-hand-rule/) says that if we extend the index finger, middle finger and thumb of our left hand in such a way that the [curren](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)t carrying conductor is placed in a [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/) (represented by the index finger) is perpendicular to the direction of [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) (represented by the middle finger), then the conductor experiences a force in the direction (represented by the thumb) mutually perpendicular to both the direction of field and the [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) in the conductor.

For clear understanding the **principle of DC motor** we have to determine the magnitude of the force, by considering the diagram below. We know that when an infinitely small charge dq is made to flow at a velocity ‘v’ under the influence of an electric field E, and a [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/) B, then the Lorentz Force dF experienced by the charge is given by:-



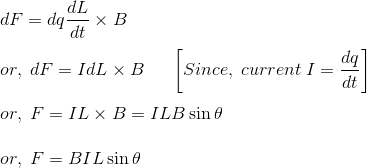
For the **operation of dc motor**, considering E = 0



i.e. it’s the cross product of dq v and [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/) B.



Where dL is the length of the conductor carrying charge q.



From the 1st diagram we can see that the [construction of a DC motor](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/) is such that the direction of [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) through the armature conductor at all instance is perpendicular to the field. Hence the force acts on the armature conductor in the direction perpendicular to the both uniform field and [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) is constant.

So if we take the [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) in the left hand side of the armature conductor to be I, and [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) at right hand side of the armature conductor to be − I, because they are flowing in the opposite direction with respect to each other.

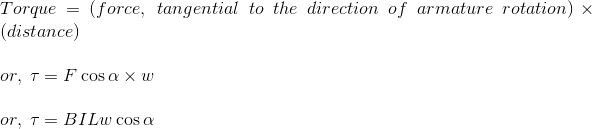
Then the force on the left hand side armature conductor,

Similarly force on the right hand side conductor

∴ we can see that at that position the force on either side is equal in magnitude but opposite in direction. And since the two conductors are separated by some distance w = width of the armature turn, the two opposite forces produces a rotational force or a torque that results in the rotation of the armature conductor.

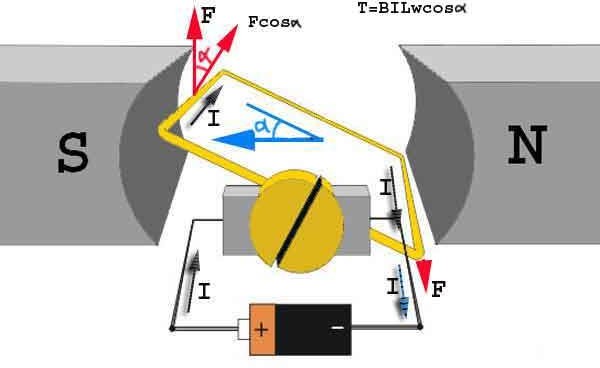
Now let's examine the expression of torque when the armature turn crate an angle of α with its initial position.

The torque produced is given by,



Where α is the angle between the plane of the armature turn and the plane of reference or the initial position of the armature which is here along the direction of [magnetic field.](http://www.electrical4u.com/what-is-magnetic-field/)

The presence of the term cosα in the torque equation very we **l** signifies that unlike force the torque at a **l** position is not the same. It in fact varies with the variation of the angle α. To explain the variation of torque and the principle behind rotation of the motor let us do a step wise analysis.

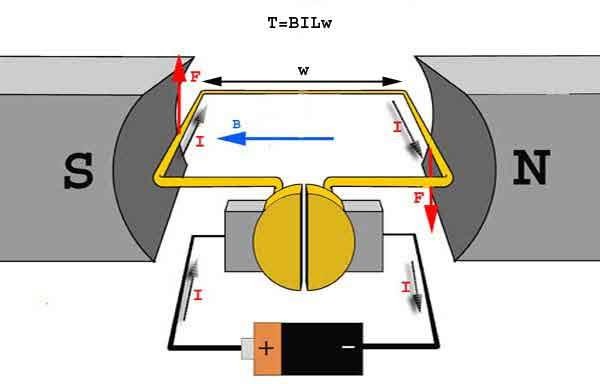


**Figure 3 permenant magnet dc motor with angle alpha**

**Step 1:** Initially considering the armature is in its starting point or reference position where the angle α = 0.



Since α = 0, the term cos α = 1, or the maximum value, hence torque at this position is maximum given by τ = BILw. This high starting torque helps in overcoming the initial inertia of rest of the armature and sets it into rotation.

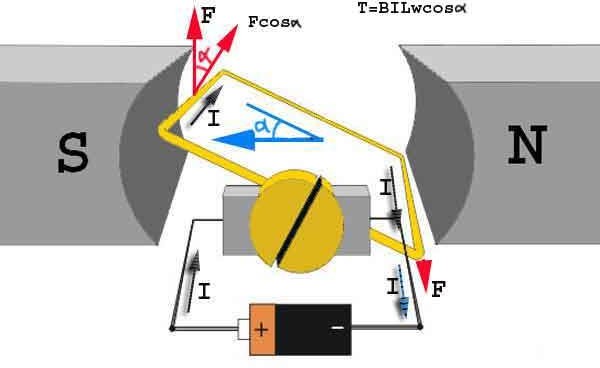


**Figure 4 permenant magnet dc motor**

**Step 2:** Once the armature is set in motion, the angle α between the actual position of the armature and its reference initial position goes on increasing in the path of its rotation until it

becomes 90° from its initial position. Consequently the term cosα decreases and also the value of torque.

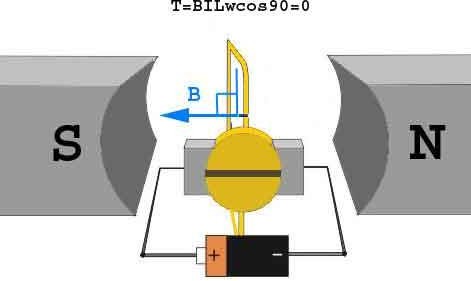
The torque in this case is given by τ = BILwcosα which is less than BIL w when α is greater than 0°.



**Step 3:** In the path of the rotation of the armature a point is reached where the actual position of the rotor is exactly perpendicular to its initial position, i.e. α = 90°, and as a result the term cosα

= 0.

The torque acting on the conductor at this position is given by,

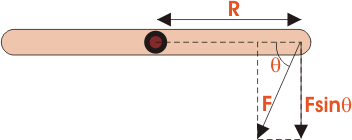


**Figure 4 permenant magnet dc motor angle 90**

i.e. virtually no rotating torque acts on the armature at this instance. But still the armature does not come to a standstill, this is because of the fact that the operation of [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) has been engineered in such a way that the inertia of motion at this point is just enough to overcome this point of null torque. Once the rotor crosses over this position the angle between the actual position of the armature and the initial plane again decreases and torque starts acting on it again.

The equation of torque is given by,

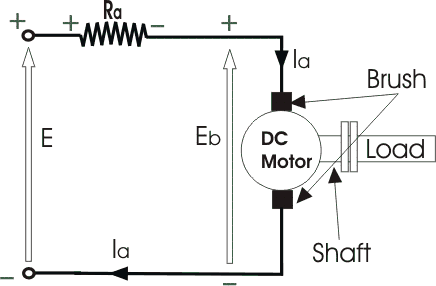
Where F is force in linear direction. R is radius of the object being rotated, and θ is the angle, the



force F is making with R vector

The [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) as we all know is a rotational machine, and **torque of dc motor** is a very important parameter in this concern, and it’s of utmost importance to understand the **torque equation of dc motor** for establishing its running characteristics.

To establish the torque equation, let us first consider the basic circuit diagram of a [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/), and



**Figure 5 dc motor circuit**

its [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) equation.

Referring to the diagram beside, we can see, that if E is the supply voltage, Eb is the back emf produced and Ia, Ra are the armatur[e curre](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)nt and armatur[e resistanc](http://www.electrical4u.com/electrical-resistance-and-laws-of-resistance/)e respectively then the [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) equation is given by,

But keeping in mind that our purpose is to derive the **torque equation of dc motor** we multiply both sides of equation (2) by Ia.



Now Ia2.Ra is the power loss due to heating of the armature coil, and the true effective mechanical power that is required to produce the desired torque of dc machine is given by,



The mechanical power Pm is related to the electromagnetic torque Tg as, Where ω is speed in rad/sec.

Now equating equation (4) & (5) we get,

Now for simplifying the torque equation of dc motor we substitute.

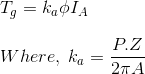


Where, P is no of poles, φ is flux per pole, Z is no. of conductors, A is no. of parallel paths, and N is the speed of the D.C. motor.

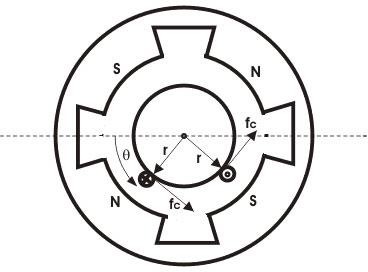


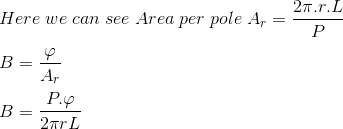
Substituting equation (6) and (7) in equation (4), we get:

The torque we so obtain, is known as the electromagnetic torque of dc motor, and subtracting the mechanical and rotational losses from it we get the mechanical torque. Therefore, Tm = Tg - mechanical losses. This is the torque equation of dc motor. It can be further simplified as:



Which is constant for a particular machine and therefore the torque of dc motor varies with only flux φ and armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) Ia. The Torque equation of a dc motor can also be explained considering the figure below



.

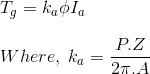
Current / conductor Ic = Ia / A Therefore, force per conductor = fc = BLIa/A Now torque Tc = fc.r

= BLIa.r/A



Hence the total torque developed of a dc machine is,

This torque equation of dc motor can be further simplified as:



Which is constant for a particular machine and therefore the torque of dc motor varies with only flux φ and armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) Ia.

# Construction of DC Motor | Yoke Poles Armature Field Winding Commutator Brushes of DC Motor

A [DC motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) like we all know is a device that deals in the conversion of electrical energy to mechanical energy and this is essentially brought about by two major parts required for the **construction of dc motor**, namely. 1) Stator – The static part that houses the field windings and receives the supply and,

1. Rotor – The rotating part that brings about the mechanical rotations. Other than that there are several subsidiary parts namely the
2. [**Yoke of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Yoke-of-DC-Motor)
3. [**Poles of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Poles-of-DC-Motor)
4. [**Field winding of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Field-Winding-of-DC-Motor)
5. [**Armature winding of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Armature-Winding-of-DC-Motor)
6. [**Commutator of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Commutator-of-DC-Motor)
7. [**Brushes of dc motor**.](http://www.electrical4u.com/construction-of-dc-motor-yoke-poles-armature-field-winding-commutator-brushes-of-dc-motor/#Brushes-of-DC-Motor)

All these parts put together configures the total **construction of a dc motor**. Now let’s do a detailed discussion about all the essential parts of dc motor.



**Figure 6 parts of dc motor**

## Yoke of DC Motor



**Figure 7 Yoke of dc motor**

The magnetic frame or the **yoke of dc motor** made up of cast iron or steel and forms an integral part of the stator or the static part of the motor. Its main function is to form a protective covering over the inner sophisticated parts of the motor and provide support to the armature. It also supports the field system by housing the magnetic poles and field winding of the dc motor.

## Poles of DC Motor

The magnetic **poles of DC motor** are structures fitted onto the inner wall of the yoke with screws. The construction of magnetic poles basically comprises of two parts namely, the pole core and the pole shoe stacked together under hydraulic pressure and then attached to the yoke. These two structures are assigned for different purposes, the pole core is of small cross sectional area and its function is to just hold the pole shoe over the yoke, whereas the pole shoe having a relatively larger cross-sectional area spreads the flux produced over the air gap between the stator and rotor to reduce the loss due to reluctance. The pole shoe also carries slots for the field windings that produce the field flux.

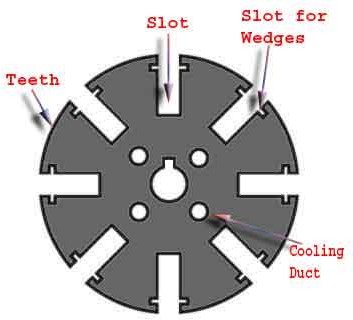
## Field Winding of DC Motor



## Figure 8 Field Winding of DC Motor

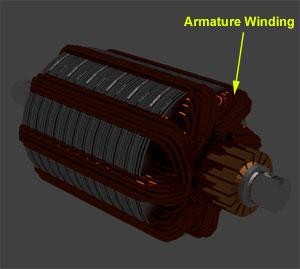
The **field winding of dc motor** are made with field coils (copper wire) wound over the slots of the pole shoes in such a manner that when field [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) flows through it, then adjacent poles have opposite polarity are produced. The field winding basically form an electromagnet, that produces field flux within which the rotor armature of the [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) rotates, and results in the effective flux cutting.

**Armature Winding of DC Motor**



The [**armature winding**](http://www.electrical4u.com/armature-winding-pole-pitch-coil-span-commutator-pitch/) **of dc motor** is attached to the rotor, or the rotating part of the machine, and as a result is subjected to altering [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/) in the path of its rotation which directly results in magnetic losses. For this reason the rotor is made of armature core, that’s made with several low-hysteresis silicon steel lamination, to reduce the magnetic losses like hysteresis and [eddy current](http://www.electrical4u.com/hysteresis-eddy-current-iron-or-core-losses-and-copper-loss-in-transformer/) loss respectively. These laminated steel sheets are stacked together to form the cylindrical structure of the armature core.

The armature core are provided with slots made of the same material as the core to which the [armature winding](http://www.electrical4u.com/armature-winding-pole-pitch-coil-span-commutator-pitch/) made with several turns of copper wire distributed uniformly over the entire periphery of the core. The slot openings a shut with fibrous wedges to prevent the conductor from plying out due to the high centrifugal force produced during the rotation of the armature, in presence of supply [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) and field.



## Figure 9 DC Motor armature winding

## Figure 9 DC Motor armature winding

The construction of **armature winding of dc motor** can be of two types:-

## Lap Winding

In this case the number of parallel paths between conductors A is equal to the number of poles P.

i.e A = P

\*\*\*An easy way of remembering it is by remembering the word LAP → L A=P

## Wave Winding

Here in this case, the number of parallel paths between conductors A is always equal to 2 irrespective of the number of poles. Hence the machine designs are made accordingly.

## commutator brushCommutator of DC Motor

## Figure 10 commutator of DC Motor

The **commutator of dc motor** is a cylindrical structure made up of copper segments stacked together, but insulated from each other by mica. Its main function as far as the dc motor is concerned is to commute or relay the supply [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) from the mains to the [armature winding](http://www.electrical4u.com/armature-winding-pole-pitch-coil-span-commutator-pitch/) housed over a rotating structure through the **brushes of dc motor**.

## Brushes of DC Motor

The **brushes of dc motor** are made with carbon or graphite structures, making sliding contact over the rotating commutator. The brushes are used to relay the [curren](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)t from external circuit to the rotating commutator form where it flows into the [armature winding.](http://www.electrical4u.com/armature-winding-pole-pitch-coil-span-commutator-pitch/) So, the commutator and brush unit of the [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) is concerned with transmitting the power from the static [electrical](http://www.electrical4u.com/electric-circuit-and-electrical-circuit-element/) [circuit](http://www.electrical4u.com/electric-circuit-and-electrical-circuit-element/) to the mechanically rotating region or the rotor.

# Armature Reaction in DC Machine

In a DC machine, the carbon brushes are always placed at the magnetic neutral axis. In no load condition, the magnetic neutral axis coincides with the geometrical neutral axis. Now, when the machine is loaded, the armature flux is directed along the inter polar axis (the axis in between the magnetic poles) and is triangular in wave shape. This results an armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) flux directed along the brush axis and causes cross magnetization of the main field. This cross magnetization effect results in the concentration of flux at the trailing pole tip in generator action and at the leading pole tip in motor action.

What is leading and trailing pole tip?

The tip of the pole from where the armature conductors come into influence is called leading tip and the other tip opposite in direction to it will be the trailing tip. For example, in the above figure if the motor rotates clockwise, then for North Pole, the lower tip is leading tip and for South Pole upper tip is leading tip. If the motion is reversed (in case of generator), the tips is interchanged. Due to cross magnetization, the magnetic neutral axis on load, shifts along the direction of rotation in DC generator and opposite to the direction of rotation in DC motor. If the brushes remain at their previous positions, then back e.m.f in case of motor or generated e.m.f in case of generator would reduce and commutation would be accompanied by heavy sparking. This is because commutation occurs at the coils located on the brushes only, and the coil undergoing commutation comes under the influence of the alternate pole(changes its location from north to south pole or vice versa). Hence, the direction of [curren](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)t flowing in the coil also reverses in a very short duration of time i.e., [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) changes from + i to – i or vice versa in a small span of time. This induces a very high magnitude of reactance [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) (L\*di/dt) in the coil which emerges out in the form of heat energy along with sparking, thus damaging the brushes and commutator segment. To reduce the adverse effects mentioned above and to improve the machine’s performance, following methods are used:

## Brush Shift

A natural solution to the problem appears to shift the brushes along the direction of rotation in generator action and against the direction of rotation in motor action, this would result into a reduction in air gap flux. This will reduce the induced [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) in generator and would increase the speed in motor. The demagnetizing m.m.f (magneto motive force) thus produced is given by: Where, Ia = armature current, Z = total number of conductors, P = total number of poles, β = angular shift of carbon brushes (in electrical Degrees). Brush shift has serious limitations, so the brushes have to be shifted to a new position every time the load changes or the direction of rotation changes or the mode of operation changes. In view of this, brush shift is limited only to very small machines. Here also, the brushes are fixed at a position corresponding to its normal load and the mode of operation. Due to these limitations, this method is generally not preferred.

## Inter Pole

The limitation of brush shift has led to the use of inter poles in almost all the medium and large sized DC machines. Inter poles are long but narrow poles placed in the inter polar axis. They have the polarity of succeeding pole(coming next in sequence of rotation) in generator action and proceeding (which has passed behind in rotation sequence) pole in motor action. The inter pole is designed to neutralize the armature reaction mmf in the inter polar axis. This is because the direction of armature reaction m.m.f is in the inter polar axis. It also provides commutation [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) for the coil undergoing commutation such that the commutation [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) completely neutralizes the reactance [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) (L di/dt). Thus, no sparking takes place.

Inter polar windings are always kept in series with armature, so inter polar winding carries the armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) ; therefore works satisfactorily irrespective of load, the direction of rotation or the mode of operation. Inter poles are made narrower to ensure that they influence only the coil undergoing commutation and its effect does not spread to the other coils. The base of the inter poles is made wider to avoid saturation and to improve response.

## Compensating Winding

Commutation problem is not the only problem in DC machines. At heavy loads, the cross magnetizing armature reaction may cause very high flux density in the trailing pole tip in generator action and leading pole tip in the motor action.

Consequently, the coil under this tip may develop induced [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) high enough to cause a flashover between the associated adjacent commutator segments particularly, because this coil is physically close to the commutation zone (at the brushes) where the air temperature might be already high due to commutation process.

This flashover may spread to the neighboring commutator segments, leading ultimately to a complete fire over the commutator surface from brush to brush. Also, when the machine is subjected to rapidly fluctuating loads, then the [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) L\* di/dt, that appears across the adjacent commutator segments may reach a value high enough to cause flashover between the adjacent commutator segments. This would start from the centre of pole as the coil below it possesses the maximum inductance. This may again cause a similar fire as described above. This problem is more acute while the load is decreasing in generating action and increasing in motor action as then, the induced e.m.f and [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) L\* di/dt will support each other. The above problems are solved by use of compensating winding.

Compensating winding consists of conductors embedded in the pole face that run parallel to the shaft and carry an armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) in a direction opposite to the direction of [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) in the armature conductors under that pole arc. With complete compensation the main field is restored. This also reduces armature circuit’s [inductor](http://www.electrical4u.com/what-is-inductor-and-inductance-theory-of-inductor/) and improves system response.Compensating winding functions satisfactorily irrespective of the load, direction of rotation and mode of operation. Obviously it is help in commutation as the inter polar winding gets relieved from its duty to compensate for the armature m.m.f under the pole arc.

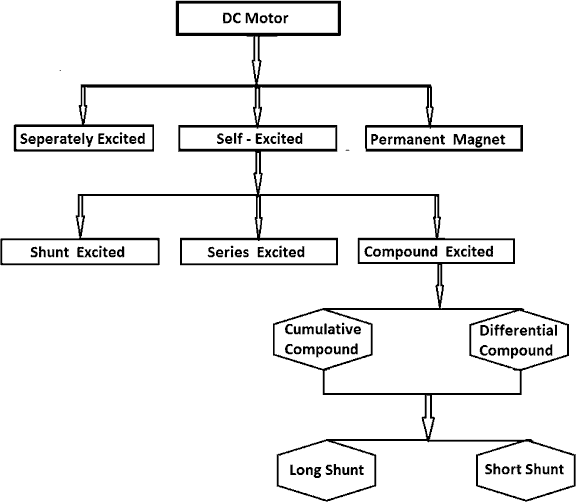
**NOTE:**

1. **The cross magnetizing armature reaction effect is mainly caused by armature conductors which are located under the pole arc. At high loads, this effect of armature reaction may cause excessive flux density in the trailing pole tip (in generator) and leading pole tip (in motor). Due to saturation in the pole shoe, the increase in flux density may be less than the reduction in the flux density in remaining section of the pole shoe. This would ultimately result into a net reduction in flux per pole. This phenomenon is thus known as the demagnetizing effect of cross magnetizing armature reaction, which is further compensated by the use of compensating windings.**
2. **Inter polar winding and compensating windings are connected in series with the armature winding but on the opposite sides with respect to armature.**
3. **The primary duty of inter polar winding is to improve the commutation process, and that of the compensating winding is to compensate for the increase or decrease in the net air gap flux i.e., to maintain its constant value.**

# Types of DC Motor Separately Excited Shunt Series Compound DC Motor

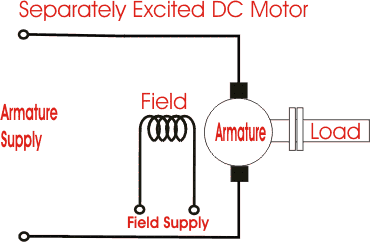
The [direct](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) [current](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) motor or the [DC motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) has a lot of application in today’s field of engineering and technology. Starting from an electric shaver to parts of automobiles, in all small or medium sized motoring applications [DC motors](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) come handy. And because of its wide range of application different functional **types of dc motor** are available in the market for specific requirements. The **types of DC motor** can be listed as follows

* + [DC motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/)
  + [Permanent Magnet DC Motor](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/)
  + Separately Excited DC Motor
  + Self Excited DC Motor
  + [Shunt Wound DC Motor](http://www.electrical4u.com/shunt-wound-dc-motor-dc-shunt-motor/)
  + [Series Wound DC Motor](http://www.electrical4u.com/series-wound-dc-motor-or-dc-series-motor/)
  + [Compound Wound DC Motor](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/)
  + Cumulative compound DC motor
  + Short shunt DC Motor
  + Long shunt DC Motor
  + Differential Compound DC Motor
  + Short Shunt DC Motor
  + Long Shunt DC Motor



Now let’s do a detailed discussion about a **l** the essential types of dc motor.

## Separately Excited DC Motor

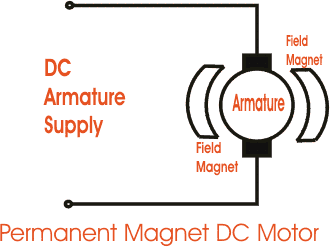


## Figure 11 separately excited DC Motor

As the name suggests, in case of a separately excited DC motor the supply is given separately to the field and armature windings. The main distinguishing fact in these types of dc motor is that, the armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) does not flow through the field windings, as the field winding is energized from a separate external source of dc [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) as shown in the figure beside.

From the [torque equation of dc motor](http://www.electrical4u.com/torque-equation-of-dc-motor/) we know Tg = Ka φ Ia So the torque in this case can be varied by varying field flux φ, independent of the armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) Ia.

## Permanent Magnet DC Motor



## Figure 12 permanent magnet DC Motor circuit

The [**permanent magnet DC motor**](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/)consists of an armature winding as in case of an usual motor, but does not necessarily contain the field windings. The construction of these types of DC motor are such that, radially magnetized permanent magnets are mounted on the inner periphery of the stator core to produce the field flux. The rotor on the other hand has a conventional dc armature with commutator segments and brushes. The diagrammatic representation of a [permanent magnet dc motor](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/) is given below. The [torque equation of dc motor](http://www.electrical4u.com/torque-equation-of-dc-motor/) suggests Tg = Ka φ Ia. Here φ is always constant, as permanent magnets of required flux density are chosen at the time of construction and can’t be changed there after. For a [permanent magnet dc motor](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/) Tg = Ka1Ia

Where Ka1 = Ka.φ which is another constant. In this case the [torque of DC Motor](http://www.electrical4u.com/torque-equation-of-dc-motor/) can only be

changed by controlling armature supply.

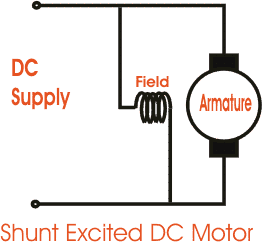
## Self Excited DC Motor

In case of self excited dc motor, the field winding is connected either in series or in parallel or partly in series, partly in parallel to the armature winding, and on this basis its further classified as:-

1. [**Shunt wound DC motor**.](http://www.electrical4u.com/shunt-wound-dc-motor-dc-shunt-motor/)
2. [**Series wound DC motor**.](http://www.electrical4u.com/series-wound-dc-motor-or-dc-series-motor/)
3. [**Compound wound DC motor**.](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/)

Let’s now go into the details of these types of self excited dc motor.

## Shunt Wound DC Motor



## Figure 13 shunt wound DC Motor

In case of a **shunt wound dc motor** or more specifically [shunt wound self excited dc motor](http://www.electrical4u.com/shunt-wound-dc-motor-dc-shunt-motor/), the field windings are exposed to the entire terminal [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) as they are connected in parallel to the armature winding as shown in the figure below.

To understand the characteristic of these types of DC motor, lets consider the basic [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) equation given by,



[Where E, Eb, Ia, Ra are the supply voltage, back emf, armature [curren](http://www.electrical4u.com/electric-current-and-theory-of-electricity/)t and armature [resistance](http://www.electrical4u.com/electrical-resistance-and-laws-of-resistance/) respectively]

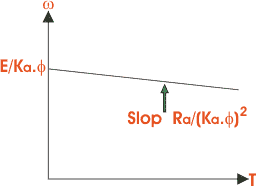


http://www.electrical4u.com/equations/tdcm1.gif[since back emf increases with flux φ and angular speed ωω] Now substituting Eb from equation (2) to equation (1) we get,

The [torque equation of a dc motor](http://www.electrical4u.com/torque-equation-of-dc-motor/) resembles,



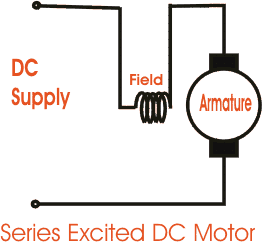
This is similar to the equation of a straight line, and we can graphically representing the torque speed characteristic of a shunt wound self excited dc motor as



The shunt wound dc motor is a constant speed motor, as the speed does not vary here with the variation of mechanical load on the output.

## Series Wound DC Motor

In case of a series wound self excited dc motor or simply **series wound dc motor**, the entire armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) flows through the field winding as its connected in series to the armature winding. The series wound self excited dc motor is diagrammatically represented below for clear understanding.



## Figure 14 series excited DC Motor

## Figure 14 series excited DC Motor

Now to determint the torque speed characteristic of these types of DC motor, lets get to the torque speed equation.

From the circuit diagram we can see that the [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) equation gets modified to Where as back emf remains Eb = kaφω

Neglecting saturation we get,

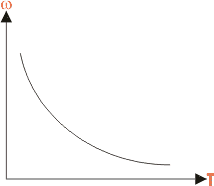
[ since field [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) = armature current]



From equation (5) & (6)

http://www.electrical4u.com/equations/tdcm3.gif

From this equation we obtain the torque speed characteristic as

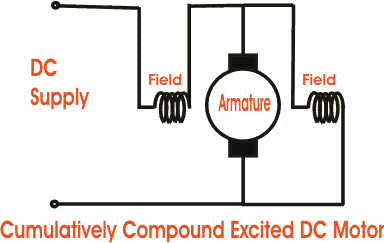


In a series wound dc motor, the speed varies with load. And operation wise this is its main difference from a shunt wound dc motor.

## Compound Wound DC Motor

The compound excitation characteristic in a [dc motor](http://www.electrical4u.com/dc-motor-or-direct-current-motor/) can be obtained by combining the operational characteristic of both the shunt and [series excited dc motor](http://www.electrical4u.com/series-wound-dc-motor-or-dc-series-motor/). The compound wound self excited dc motor or simply [**compound wound dc motor**](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/)essentially contains the field winding connected both in series and in parallel to the armature winding as shown in the figure below

## Figure 15 cumulatively compound excited DC Motor



:

The excitation of [compound wound dc motor](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/) can be of two types depending on the nature of compounding.

## Cumulative Compound DC Motor

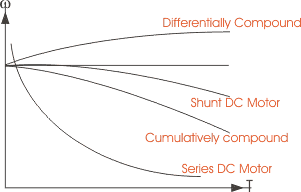
When the shunt field flux assists the main field flux, produced by the main field connected in series to the armature winding then its called cumulative compound dc motor.

## Differential Compound DC Motor

In case of a differentially compounded self excited dc motor i.e. differential compound dc motor, the arrangement of shunt and series winding is such that the field flux produced by the shunt field winding diminishes the effect of flux by the main series field winding.

The net flux produced in this case is lesser than the original flux and hence does not find much of a practical application.

The compounding characteristic of the self excited dc motor is shown in the figure below.



## Figure 14 torque and omega graph

Both the cumulative compound and differential compound dc motor can either be of short shunt or long shunt type depending on the nature of arrangement.

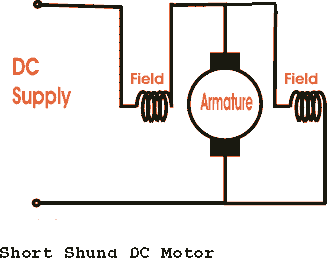
## Short Shunt DC Motor

If the shunt field winding is only parallel to the armature winding and not the series field winding then its known as short shunt dc motor or more specifically short shunt type [compound](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/) [wound dc motor.](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/)

## Long Shunt DC Motor

If the shunt field winding is parallel to both the armature winding and the series field winding then it’s known as long shunt type [compounded wound dc motor](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/) or simply long shunt dc motor.

Short shunt and long shunt type motors have been shown in the diagram below.



## Figure 16 short shunt DC Motor

# Characteristics Of DC Motors

Generally, three characteristic curves are considered for [DC motors](http://www.electricaleasy.com/2014/01/basic-working-of-dc-motor.html) which are, (i) Torque vs. armature current (Ta - Ia), (ii) Speed vs. armature current and (iii) Speed vs. torque. These are explained below for each [type of DC motor.](http://www.electricaleasy.com/2012/12/classifications-of-dc-machines.html) These characteristics are determined by keeping following two relations in mind.

Ta α Φ.Ia and N α Eb/Φ

## Characteristics Of DC Series Motors Torque Vs. Armature Current (Ta-Ia)

This characteristic is also known as **electrical characteristic**. We know that torque is directly proportional to armature current and flux, Ta α Φ.Ia. In DC series motors, field winding is connected in series with armature. Thus, before magnetic saturation of the field, flux Φ is directly proportional to Ia. Therefore, before magnetic saturation Ta α Ia2. At light loads, Ia as we **l** as Φ is sma **l** and hence the torque increases as the square of the armature current.

Therefore, the Ta-Ia curve is parabola for smaller values of Ia.

After magnetic saturation of the field winding, flux Φ is independent of armature current Ia. Therefore, the torque varies proportional to Ia only, T α Ia.Therefore, after magnetic saturation, Ta-Ia curve becomes straight line.

The shaft torque (Tsh) is less than armature torque (Ta) due to [stray losses.](http://www.electricaleasy.com/2014/01/losses-in-dc-machine.html)

In DC series motors, (prior to magnetic saturation) torque increases as the square of armature current, these motors are used where high starting torque is required

## Speed Vs. Armature Current (N-Ia)

We know the relation, N α Eb/Φ

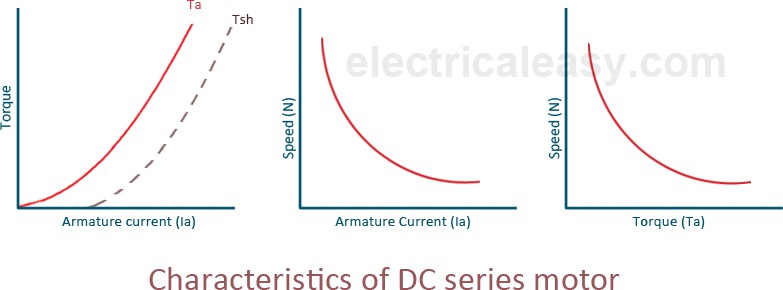
For small load current (and hence for small armature current) change in back emf Eb is small and it may be neglected. Thus, for sma **l** currents speed is inversely proportional to Φ. As we know, flux is directly proportional to Ia, speed is also inversely proportional to Ia.

When armature current is very small the speed becomes dangerously high. That is why **a series motor should never be started without some mechanical load**.

But, at heavy loads, armature current Ia is large. And hence speed is low which results in decreased back emf Eb. Due to decreased Eb, more armature current is allowed.

## Speed Vs. Torque (N-Ta)

This characteristic is also called as **mechanical characteristic**. From the above two characteristics of DC series motor, it can be found that when speed is high, torque is low and vice versa.

[](http://3.bp.blogspot.com/-cUu8u-bZm0o/U7qEkWngbRI/AAAAAAAAA58/-7F6w5isurc/s1600/Characteristics+of+DC+series+motor.png)

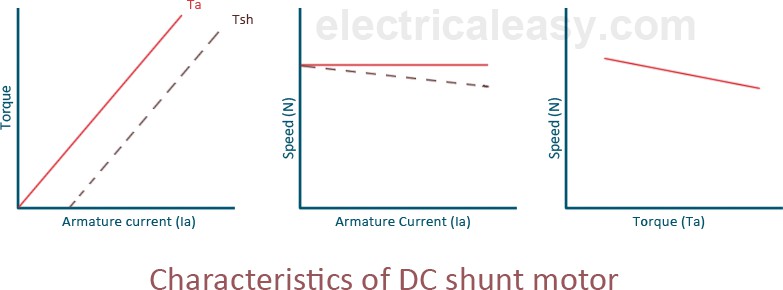
## Characteristics Of DC Shunt Motors Torque Vs. Armature Current (Ta-Ia)

In case of DC shunt motors we can assume the field flux Φ to be constant. Though at heavy loads, Φ decreases in a sma **l** amount due to increased [armature reaction.](http://www.electricaleasy.com/2013/01/armature-reaction-in-dc-machines.html) But as we are neglecting the change in the flux Φ, we can say that torque is proportional to armature current. Hence the Ta-Ia characteristic for a dc shunt motor will be a straight line through origin.

Since, heavy starting load needs heavy starting current, **shunt motor should never be started on a heavy load**.

## Speed Vs. Armature Current (N-Ia)

As flux Φ is assumed constant, we can say N α Eb. But, back emf is also almost constant, the speed remains constant. But practica **l**y, Φ as we **l** as Eb decreases with increase in load. But, the Eb decreases slightly more than Φ, and hence the speed decreases slightly. Genera **l**y, the speed decreases by 5 to 15% of full load speed only. And hence, **a shunt motor can be assumed as a constant speed motor**.

[](http://1.bp.blogspot.com/-C_FMlkL4L94/U7qF335Ne3I/AAAAAAAAA6E/PkKtiuwDtaI/s1600/Characteristics+of+DC+shunt+motor.png)

## Characteristics Of DC Compound Motor

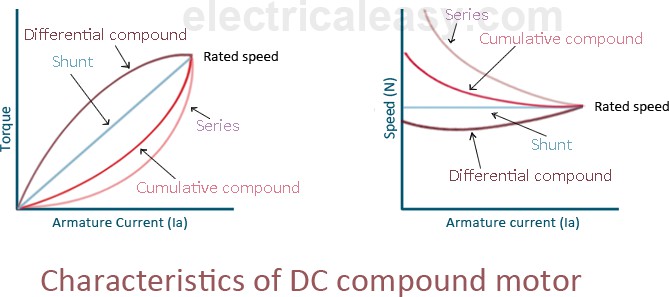
DC compound motors have both series as well as shunt windings. In a compound motor series and shunt windings are connected such that series flux is in direction with shunt flux then the motor is said to be cumulatively compounded. And if series flux is opposite direction as that of the shunt flux, then the motor is said to be differentially compounded. Characteristics of both these types are explained below.

## Cumulative compound motor

Cumulative compound motors are used where series characteristics are required but the load is likely to be removed completely. Series winding takes care of the heavy load, whereas the shunt winding prevents the motor from running at dangerously high speed when the load is suddenly removed. These motors are generally employed a flywheel, where sudden and temporary loads are applied like in rolling mills.

## Differential compound motor

Since in differential field motors, series flux opposes shunt flux, the total flux decreases with increase in load. Due to this, the speed remains almost constant or even it may increase slightly with increase in load. Differential compound motors are not commonly use, but they find limited applications in experimental and research work.

[](http://4.bp.blogspot.com/-cs_O6WoxO20/U7qGoSQmGnI/AAAAAAAAA6M/Q-TgOKMeQbU/s1600/Characteristics+of+DC+compound+motor.png)

# Speed Regulation of DC Motor

On application of load the speed of a dc motor decreases gradually. This is not at all desirable. So the difference between no load and full load speed should be very less. The motor capable of maintaining a nearly constant speed for varying load is said to have good speed regulation i.e the difference between no load and full load speed is quite less. The speed regulation of a [permanent](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/) [magnet DC motor](http://www.electrical4u.com/permanent-magnet-dc-motor-or-pmdc-motor/) is good ranging from 10 - 15% whereas for [dc shunt motor](http://www.electrical4u.com/shunt-wound-dc-motor-dc-shunt-motor/) it is somewhat less than 10 %. [DC series motor](http://www.electrical4u.com/series-wound-dc-motor-or-dc-series-motor/) has poor value of regulation. In case of [compound DC motor](http://www.electrical4u.com/compound-wound-dc-motor-or-dc-compound-motor/) the speed regulation for dc cumulative compound is around 25 % while differential compound has its excellent value

Speed of a DC Motor

The emf equation of DC motor is given by

Here N = speed of rotation in rpm. P = number of poles. A = number of parallel paths. Z = total no. conductors in armature.



Hence, speed of a DC motor is directly proportional to emf of rotation (E) and inversely proportional to flux per pole (φ).

Speed Regulation of a DC Motor

The speed regulation is defined as the change in speed from no load to full load, expressed as a fraction or percentage of full load speed. Therefore, as per definition per unit (p.u) **speed regulation of DC motor** is given as,

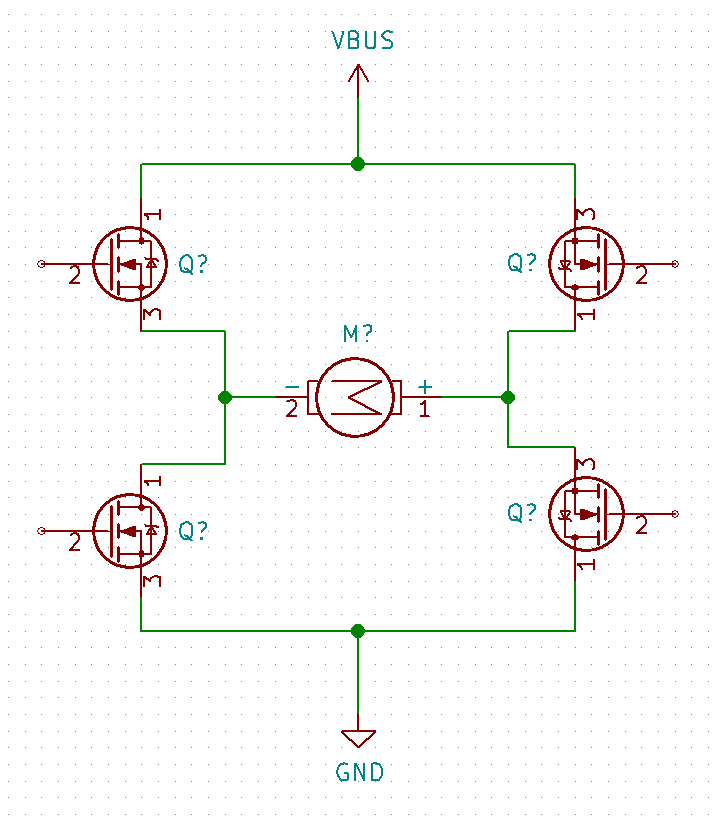
Similarly, percentage (%) **speed regulation** is given as,



Where Nno load = no load speed and Nfull lod = full load speed of DC motor. Therefore, Percent speed regulation = Per unit (p.u) speed regulation X 100 %. A motor which has nearly constant speed at all load below full rated load, have good speed regulation.

**H BRIDGE**

**H BRIDGE DESCRIPTION AND APPLICATIONS:**

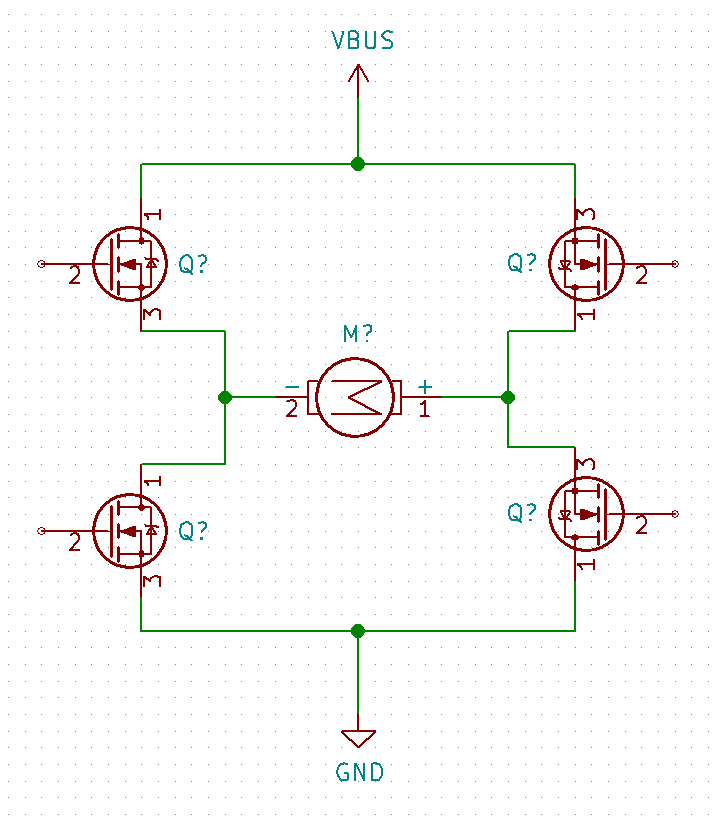
that an arbitrary load impedance is decoupled from a direct current (DC) power rail and ground. These switches can then be used to control the direction of current running from the DC source to ground in either direction across the connected impedance. The “H” in H Bridge comes from the shape of the bridge, where either side of the H is different two switches in series between the DC rail and ground while the centerline of the H is an arbitrary impedance. An example of a simple H Bridge with four switches and single load impedance is shown in Figure 1 to the right. Each of the switches in this figure are independent from each other and only have two positions, either conducting current (ON) or blocking current (OFF).

## Figure 17 H Bridge circuit

H Bridges can be found in many different applications where there is a desire to have control over the direction current can flow. Some common examples of this would be controlling the direction an electric motor can turn by allowing current to flow in one direction and then reversing that direction in the bridge, thus causing the motor to turn in the opposite direction. In the case of the high-powered inverter being constructed by Team 7 of ECE 480 Fall Semester 2014, an H Bridge is being utilized to create an Alternating Current (AC) waveform from a high voltage DC Rail. This is done by reversing the direction of current flow across the load impedance at a frequency of 60 Hz which in turn results in an alternating current signal at the same frequency of typical line current in the United States.

**BASIC H BRIDGE DESIGN**

Each of the switches shown in have different roles for typical operation of an H Bridge. The first important distinction between the different switches within the circuit is that the top two switches are referred to as the **High Side** and the bottom two switches are referred to as the **Low Side.** This clarification is important to the design of an H Bridge because the functionality of these sides varies based on the application that the bridge is being used in. The high side switches are responsible for controlling the availability of the DC Rail voltage across the load impedance while the low side switches are responsible for controlling the connection between load impedance and ground.

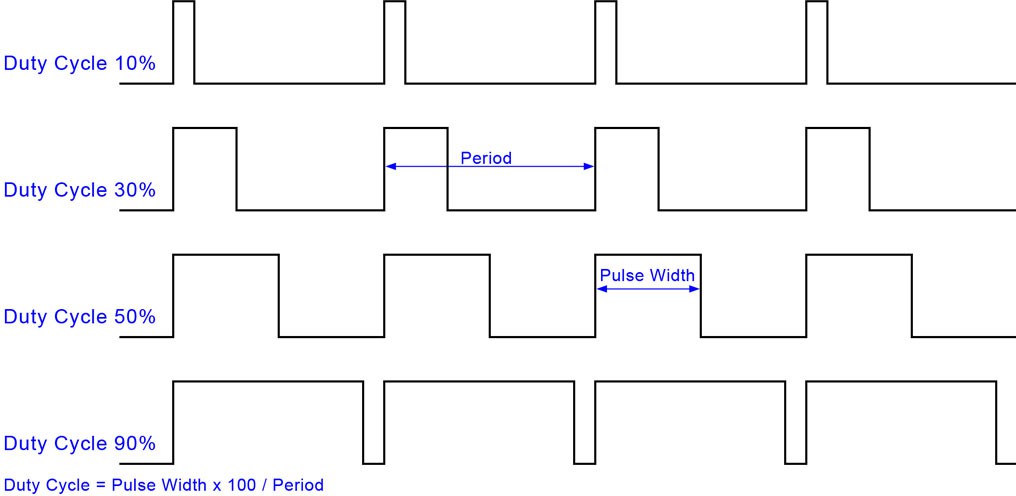


## Figure 17 H Bridge circuit

The next important clarification to make in designing an H Bridge is what type of components will be used to act as the four switches within the final circuit design.Most applications of H Bridges use four Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) to act as voltage-controlled switches. These MOSFETs can be either P- Channel or N-Channel depending on the design requirements for any specific application.

**PULSE WIDTH MODULATION**

The final piece required in understanding H Bridge design is the type of signals that can be supplied to the bridge. The MOSFETs in the bridge will only react to either a high (ON) or low (OFF) signal, meaning all signals run to the gate drivers must be a mixture of these two states. The easiest way to go about creating different waveforms with this constraint is to use a form of modulation called Pulse Width Modulation (PWM). PWM is created by allocating a slot of time and then varying the amount of time the signal is on compared to off within that originally determined slot. The signal can be off or on anywhere from 0% to 100% of the time, as long as the percentage of both the off time added to the percentage of on time is equal to 100%. This percentage of time on to time off is called the Duty Cycle of the signal. Once the PWM signal is created it can be fed into the Gate Drivers that will then amplify the high side and low side signals in order to meet the VGS(th) requirement of each driver within the H Bridge configuration. An example of several PWM control signals is shown in Figure 4 6 below.



**Encoders**

**Theory and Methods, and Signal Output Styles of** **Encoders :**

Encoders are used to determine the position, velocity and direction of a motor shaft or other mechanical motion. They provide information required for the precise control of a variety of applications, such as positioning a rotary table, pick and place, machine assembly, packaging, robotics and more. Regardless of type, all encoders provide a method of orientation detection that’s used as a reference point for position control.

An encoder (for industrial controls) is a special sensor that captures position information and relays that data to other devices. The position information can be determined using one of three technologies: optical, magnetic or capacitive.

Optical encoders are the most accurate of the standard styles of encoders. When choosing an optical encoder, it’s important that the encoder has extra protections built in to prevent contamination from dust, vibration and other conditions common to industrial environments.

Magnetic encoders are more rugged than optical encoders and often used in environments in which dirt, steam, vibration and other factors could interfere with optical encoder performance. Magnetic encoders cannot achieve the resolution or accuracy of optical encoders.

Capacitive encoders are relative newcomers as an industrial sensing technology. These encoders are as rugged as magnetic encoders but also do not achieve the high resolutions of optical encoders.

Regardless of the sensing technology an encoder employs, the encoder’s electronics sense movement and translate that motion into industry-standard electrical signals

**Rotary vs. Linear**

There are two basic geometries for encoders: linear and rotary. Rotary and linear encoders work in similar ways. As the names imply, linear encoders measure motion along a path, and rotary encoders identify rotational motion. Thus the application determines which encoder is best suited for the job.

A linear encoder typically consists of a scale (a coded strip) and a sensing “head” that reads the spacing between the scales’ coding to determine position. A linear encoder’s resolution is measured in pulses per distance (pulses per inch (ppi), pulses per millimeter, etc.). The scale (coded strip) has a set resolution with marks embedded into it or on it, which is read by the head. A linear encoder with 100 ppi resolution would read 100 marks for every one inch of movement.

Unlike linear encoder resolution measurement, rotary encoder resolution is measured in pulses per revolution (PPR), also known as “line count.” A rotary encoder is commonly comprised of an internal coded disc and a sensing head used to read rotary position. A linear encoder is very similar to a tape measure, while a rotary encoder is more like a measuring wheel. A rotary encoder with 100 ppr resolution would read 100 marks on its coded disc for every revolution.

**Incremental vs. Absolute**

Encoders come in incremental and absolute styles. Like linear and rotary encoders, incremental and absolute encoders have similarities, but they differ in wiring and movement identification.

An incremental encoder only reads pulses to provide information about the relative motion of the shaft. It has no information about location when powered up; it can only show how far the shaft has moved since the encoder was powered up. It reports back these position changes with electrical “pulses”. These pulse streams can either be single channel (one output wire from the encoder) or dual channel (two wires – see also the “Quadrature” section below). Think of an incremental encoder as a tape measure with no numbers on it, only tick marks: you can tell how far you’ve moved, but you don’t know exactly where you are unless you measure from a known spot

**What Is Quadrature Output?**

Quadrature output utilizes two different sets of “slots” or channels (A and B) on the optical disc inside the encoder case, separated by 90 degrees of phase shift (Image 1). These two outputs can either be ON or OFF, resulting in four different “states” for each segment of resolution. The image below shows the four different states of the quadrature output.

Time slice “a”: A = OFF and B = ON Time slice “b”: A and B both OFF Time slice “c”: A = On and B = OFF Time slice “d”: A and B both ON.

Normal revolution (CW)

T(100%)

a b c d

H

OUT A

L

H

OUT B

L

The output timing diagram for a quadrature output shows A = ON, then B = ON when rotation is in a clockwise direction.

Therefore, a quadrature encoder with a resolution of 100 pulses per revolution (100 “slots” of an A channel or B channel) would actually produce 400 different states for each revolution of the encoder. That’s why quadrature encoders are sometimes referred to as x4 (times 4) encoders. The pattern of A and B turning ON and OFF also reveals which direction the encoder is turning. The encoder diagram above has A = ON, then B = ON when rotated in the clockwise direction. If this encoder were rotated counter clockwise, B would turn ON first, then A would turn ON. Encoders may differ in their definition of direction based on quadrature pattern.

**Ultrasonic sensor**

Ultrasonic sensors paintings with the aid of using sending out a valid wave at a frequency above the variety of human hearing. The transducer of the sensor acts as a microphone to acquire and ship the ultrasonic sound. Our ultrasonic sensors, like many others, use an unmarried transducer to ship a pulse and to acquire the echo.

The sensor determines the space to a goal with the aid of using measuring time lapses among the sending and receiving of the ultrasonic pulse. The running precept of this module is simple. It sends an ultrasonic pulse out at forty kHz which travels via the air and if there's an impediment or object, it'll get better to the sensor.

By calculating the tour time and the rate of sound, the space may be calculated. Ultrasonic sensors are a superb answer for the detection of clean items. For liquid stage measurement, packages that use infrared sensors, for instance, warfare with this specific use case due to goal translucence. For presence detection, ultrasonic sensors hit upon items irrespective of the color, surface, or cloth (until the cloth could be very smooth like wool, as it'd soak up sound )To hit upon obvious and different gadgets wherein optical technology may also fail, ultrasonic sensors are a dependable choice.

Ultrasonic distance, stage, and proximity sensors are typically used with microcontroller structures like Raspberry Pi, ARM, PIC, Arduino, Beagle Board, and more. Ultrasonic sensors transmit sound waves in the direction of a goal and could decide its distance with the aid of using measuring the time it took for the contemplated waves to go back to the receiver.

This sensor is a digital tool with a view to degree the space of a goal with the aid of using transmitting ultrasonic sound waves, after which will convert the contemplated sound into an electrical signal. Ultrasonic sensors are also used in obstacle avoidance systems, as well as in manufacturing. In some cases, the target object is so small that the reflected ultrasonic signal is insufficient for detection, and the distance cannot be measured correctly. Ultrasound is reliable in any lighting environment and can be used inside or outside.  Ultrasonic sensors can handle collision avoidance for a robot and are being moved often, as long as it isn’t too fast.  Ultrasonic are so widely used, they can be reliably implemented in grain bin sensing applications, water level sensing, drone applications, and sensing cars at your local drive-thru restaurant or bank. Ultrasonic rangefinders are commonly used as devices to detect a collision

### 

### Ultrasonic Sensors are best used in the non-contact detection of:

* Presence
* Level
* Position
* Distance
* Non-contact sensors are also referred to as **proximity sensors.**

**Ultrasonic sensors are Independent of:**

* light
* Smoke
* Dust
* Color
* Material (except for soft surfaces, i.e. wool, because the surface absorbs the ultrasonic sound wave and doesn’t reflect sound.)

**PID Controller**

A **proportional-integral-derivative controller** (**PID controller**) is a control loop feedback mechanism (controller) widely used in industrial control systems. A PID controller calculates an *error* value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the *error* by adjusting the process through use of a manipulated variable.

The PID controller algorithm involves three separate constant parameters, and is accordingly sometimes called **three-term control**: the proportional, the integral and derivative values, denoted *P, I,* and *D.* Simply put, these values can be interpreted in terms of time: ***P* depends on the *present* error, *I* on the accumulation of *past* errors, and *D* is a prediction of *future* errors**, based on current rate of change. The weighted sum of these three actions is used to adjust the process via a control element such as the position of a control valve, a damper, or the power supplied to a heating element.

In the absence of knowledge of the underlying process, a PID controller has historically been considered to be the most useful controller. By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint, and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability.

Some applications may require using only one or two actions to provide the appropriate system control. This is achieved by setting the other parameters to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are fairly common, since derivative action is sensitive to measurement noise, whereas the absence of an integral term may prevent the system from reaching its target value due to the control action.

# PID Controller Theory

The PID control scheme is named after its three correcting terms, whose sum constitutes the manipulated variable (MV). The proportional, integral, and derivative terms are summed to calculate the output of the PID controller. Defining (𝑡) as the controller output,

Controller manufacturers arrange the Proportional, Integral and Derivative modes into three different controller algorithms or controller structures. These are called

Interactive, Noninteractive, and Parallel algorithms. Some controller manufacturers allow you to choose between different controller algorithms as a configuration option in the controller software. The PID Algorithms are:

* 1. **Interactive Algorithm**

1 𝑡 𝑑

(𝑡) = 𝐾𝑐 [(𝑡) + 𝑇𝑖 ∫ 𝑒(𝑟)𝑑𝑟] × [1 + 𝑇𝑑 𝑑𝑡 𝑒(𝑡)]

0

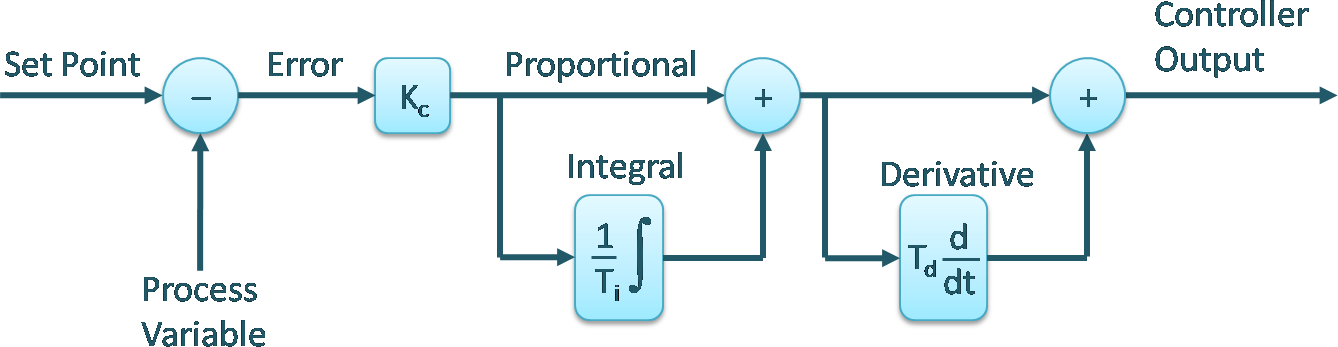


Figure 18: Interactive Algorithm

* 1. **NonInteractive Algorithm**

𝑢(𝑡) = 𝐾𝑐 [𝑒(𝑡) + 𝑇𝑖 ∫ 𝑒(𝑟)𝑑𝑟 + 𝑇𝑑 𝑑𝑡 𝑒(𝑡)]

0

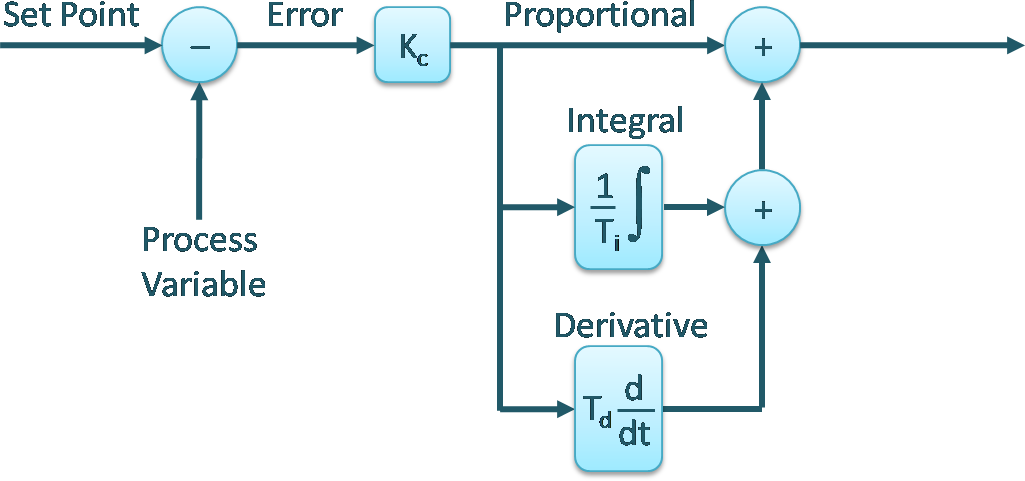


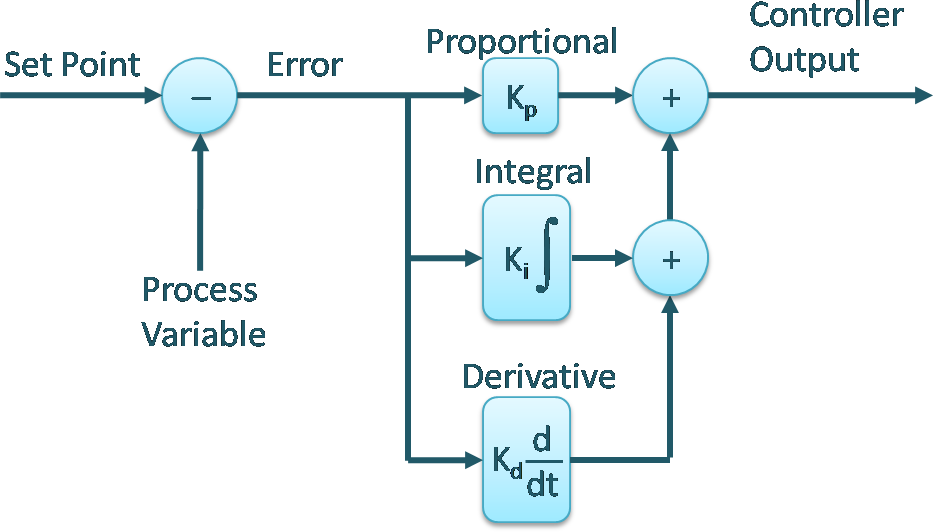
Figure 19: NonIneractive Algorithm

Parallel Algorithm

𝑡 𝑑

## 𝑢(𝑡) = 𝐾𝑝𝑒(𝑡) + 𝐾𝑖 ∫ 𝑒(𝑟)𝑑𝑟 + 𝐾𝑑 𝑑𝑡 𝑒(𝑡)

0



Where

𝐾𝑝 = 𝐾𝑐: 𝑃𝑟𝑜𝑝𝑜𝑡𝑖𝑜𝑛𝑎𝑙 𝐺𝑎𝑖𝑛

Figure 20: Parallel Algorithm

𝐾𝑖

= 𝐾𝑐 : 𝐼𝑛𝑡𝑒𝑔𝑟𝑎𝑙 𝐺𝑎𝑖𝑛

𝑇𝑖

𝐾𝑑 = 𝐾𝑐𝑇𝑑: 𝐷𝑒𝑟𝑖𝑣𝑎𝑡𝑖𝑣𝑒 𝐺𝑎𝑖𝑛

𝑒(𝑡) = 𝑟(𝑡) − 𝑦(𝑡)

#### Proportional Term

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant Kp, called the proportional gain constant. The proportional term is given by:

𝑃𝑜𝑢𝑡 = 𝐾𝑝𝑒(𝑡)

A high proportional gain results in a large change in the output for a given change in the error. If the proportional gain is too high, the system can become unstable. In contrast, a small gain results in a small output response to a large input error, and a less responsive or less sensitive controller. If the proportional gain is too low, the control action may be too small when responding to system disturbances. Tuning theory and industrial practice indicate that the

proportional term should contribute the bulk of the output change

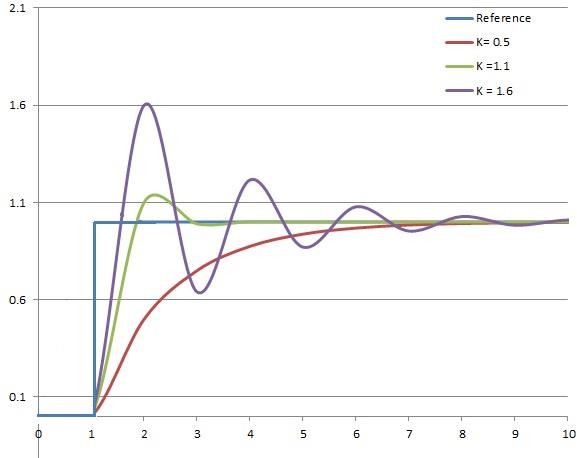


Figure 21: The effect of add 𝐾𝑝 (𝐾𝑖, 𝑎𝑛𝑑 𝐾𝑑) held constant

#### Integral Term

The contribution from the integral term is proportional to both the magnitude of the error and the duration of the error. The [integral](http://en.wikipedia.org/wiki/Integral) in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The accumulated error is then multiplied by the integral gain 𝑲𝒊 and added to the controller output.

𝑡

𝐼𝑂𝑢𝑡 = 𝐾𝑖 ∫ 𝑒(𝑟)𝑑𝑟

0

The integral term accelerates the movement of the process towards set-point and eliminates the residual steady-state error that occurs with a pure proportional controller. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the set-point value.

#### Derivative Term

The [derivative](http://en.wikipedia.org/wiki/Derivative) of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain 𝐾d. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain, 𝐾d. The derivative term is given by 𝐷𝑜𝑢𝑡 = 𝐾𝑑 𝑑𝑡 (𝑡)

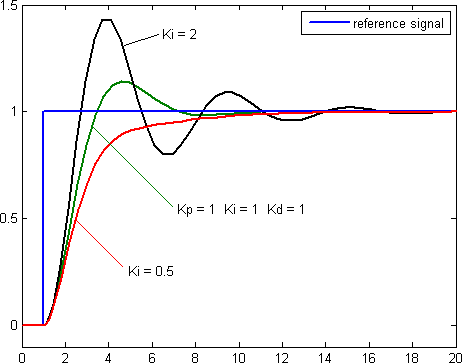


Figure 22: The effect of add 𝐾i (𝐾p, 𝑎𝑛𝑑 𝐾d) held constant

Derivative action predicts system behavior and thus improves settling time and stability of the system. An ideal derivative is not [causal](http://en.wikipedia.org/wiki/Causal_system), so that implementations of PID controllers include an additional low pass filtering for the derivative term, to limit the high frequency gain and noise. Derivative action is seldom used in practice though - by one estimate in only 20% of deployed controllers- because of its variable impact on system stability in real-world applications.

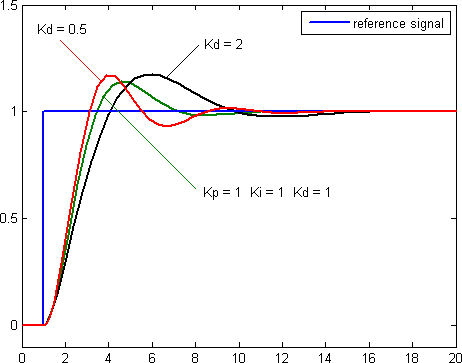


Figure 23 The effect of add 𝐾d (𝐾p, 𝑎𝑛𝑑 𝐾i) held constant

Table 1: Effect of increasing parameter independently

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameter** | **Rise Time** | **Overshoot** | **Settling Time** | **Steady-State Error** | **Stability** |
| 𝑲𝒑 | Decrease | Increase | Small Change | Decrease | Degrade |
| 𝑲𝒊 | Decrease | Increase | Increase | Eliminate | Degrade |
| 𝑲𝒅 | Minor Change | Decrease | Decrease | No Effect | Improve if  𝐾𝑑 small |

# Overview of Methods

There are several methods for tuning a PID loop. The most effective methods generally involve the development of some form of process model, then choosing P, I, and D based on the dynamic model parameters. Manual tuning methods can be relatively inefficient, particularly if the loops have response times on the order of minutes or longer.

The choice of method will depend largely on whether or not the loop can be taken "offline" for tuning, and on the response time of the system. If the system can be taken offline, the best tuning method often involves subjecting the system to a step change in input, measuring the output as a function of time, and using this response to determine the control parameters.

Table 2: Choosing a Tuning Method

|  |  |  |
| --- | --- | --- |
| **Method** | **Advantages** | **Disadvantages** |
| **Manual Tuning** | No math required , Online | Requires experienced personnel |
| **Ziegler-Nichols** | Proven Method, Online | Process upset, some trial-and- error, very aggressive tuning |
| **Cohen-Coon** | Good process models | Some math; offline; only good for first-order processes |
| **Software Tools** | Consistent tuning; online or offline - can employ computer-automated control system design ([CAutoD](http://en.wikipedia.org/wiki/CAutoD)) techniques; | Some cost or training involved |

# Open Loop Method

In these methods, the PID is being tuned in open loop, isolated from the process plant. First a step input is applied to the plant and the process reaction curve is obtained. Using the process reaction curve with one of the First Order Plus Dead Time (FOPDT) estimation methods an approximation of the process is calculated. Knowing 𝐾𝑚, 𝑟𝑚 and 𝑡𝑑 the PID parameters can be evaluated from the related correlations according to the method used.

First Order Plus Dead Time (FOPDT) is given by

𝐺(𝑠) = 𝐾𝑚 𝑒−𝑡𝑑𝑠

𝑟𝑚𝑠 + 1

### Ziegler-Nichols Open Loop Method

In the 1940's, Ziegler and Nichols devised two empirical methods for obtaining controller parameters. Their methods were used for first order plus dead time situations, and involved intense manual calculations. With improved optimization software, most manual methods such as these are no longer used. However, even with computer aids, the following two methods are still employed today, and are considered among the most common.

This method remains a popular technique for tuning controllers that use proportional, integral, and derivative actions. The Ziegler-Nichols open-loop method is also referred to as S-shaped curve method, because it tests the open-loop reaction of the process to a change in the control variable output. This basic test requires that the response of the system be recorded, preferably by a plotter or computer. Once certain process response values are found, they can be plugged into the Ziegler- Nichols equation with specific multiplier constants for the gains of a controller with either P, PI, or PID actions.

In this method, we obtain experimentally the open loop response of the FOPDT to a unit step input. This method only applied if the response to a step input exhibits an *s*-shaped curve as shown in figure 7. This means that if the plant involves integrators (like 2nd order prototypes system) or complex-conjugate poles (general 2nd order system), then this method can’t be applied since *s*- shaped will not be obtained.

This method remains a popular technique for tuning controllers that use proportional, integral, and derivative actions. The Ziegler-Nichols open-loop method is also referred to as a process reaction method, because it tests the open-loop reaction of the process to a change in the control variable output.

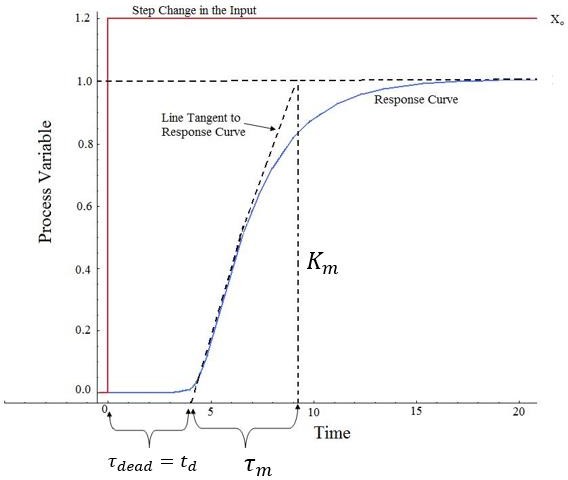
#### The Tuning Procedure:

To use the Ziegler-Nichols open-loop tuning method, you must perform the following steps:

1. Make an open loop step test

From the [process reaction curve](http://controls.engin.umich.edu/wiki/index.php/PIDTuningClassical#Process_Reaction_Curve) determine the transportation lag or dead time, 𝑡𝑑, the time constant or time for the response to change, 𝑟𝑚, and the ultimate value that the response reaches at steady-state, 𝐾𝑚, for a step change of 𝑋𝑜.

1. Determine the loop tuning constants. Plug in the reaction rate and lag time values to the Ziegler-Nichols open-loop tuning equations for the appropriate controller (P, PI, or PID) to calculate the controller constants. Use the table 3.



**Figure 24:** Open Loop of First order system plus dead Time (s-shaped curve)

**Table 3**: Open-loop Calculation of (𝐾𝑝. 𝑇𝑖. 𝑇𝑑)

|  |  |  |  |
| --- | --- | --- | --- |
|  | 𝑲𝒑 | 𝑻𝒊 | 𝑻𝒅 |
| **P- Controller** | 𝑋𝑜 𝑟𝑚  𝐾𝑚 𝑡𝑑 | ∞ | 0 |
| **PI- Controller** | 𝑋𝑜 𝑐𝑚  0.9  𝐾𝑚 𝑡𝑑 | 3.3 𝑡𝑑 | 0 |
| **PID- Controller** | 𝑋𝑜 𝑐𝑚  1.2  𝐾𝑚 𝑡𝑑 | 2 𝑡𝑑 | 0.5 𝑡𝑑 |

* This mean that the controller adds double zero at 𝑠 = − 1 , and pole at origin

𝑡𝑑

* Advantages Ziegler-Nichols Open Loop Tuning Methods

1. Quick and easier to use than other methods
2. It is a robust and popular method

* Disadvantages Ziegler-Nichols Open Loop Tuning Methods

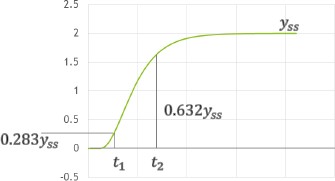
1. It depends upon purely 𝑡𝑑to estimate I and D controllers.
2. Approximations for the 𝐾𝑝. 𝑇, and 𝑇𝑑 values might not be entirely accurate for different systems.
3. It does not hold for I, D and PD controllers

### Cohen-Coon Method

The Cohen-Coon tuning rules are suited to a wider variety of processes than the Ziegler-Nichols tuning rules. The Ziegler-Nichols rules work well only on processes where the dead time is less than half the length of the time constant. The Cohen-Coon tuning rules work well on processes where the dead time is less than two times the length of the time constant (and you can stretch this even further if required). Also it provides one of the few sets of tuning rules that has rules for PD controllers.

Like the Ziegler-Nichols tuning rules, the Cohen-Coon rules aim for a quarter-amplitude damping response. Although quarter-amplitude damping-type of tuning provides very fast disturbance rejection, it tends to be very oscillatory and frequently interacts with similarly-tuned loops. Quarter-amplitude damping-type tuning also leaves the loop vulnerable to going unstable if the process gain or dead time doubles in value.

In this method the process response curve is obtained first, by an open loop test as shown in figure8 and then the process dynamics is approximated by a first order plus dead time model, with following parameters:



**Figure 25:** The open loop response of plant

3

𝑟𝑚 = 2 (𝑡2 − 𝑡1)

𝑡𝑑 = 𝑡2 − 𝑟𝑚

Again the particular rules for this method are used to calculate the PID parameters. They are listed in table 4

**Table 4:** the parameter of Cohen-Coon Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | 𝑲𝒄 | 𝑻𝒊 | 𝑻𝒅 |
| **P** | 𝑟𝑚 𝑡𝑑  𝐾𝑡 (1 + 3𝑟 )  𝑑 𝑚 | - | - |
| **PI** | 𝑟𝑚 𝑡𝑑  𝐾𝑡 (0.9 + 12𝑟 )  𝑑 𝑚 | 30 + 3𝑡𝑑  𝑟𝑚  𝑡𝑑 ( 20𝑡𝑑) 9 + 𝑟𝑚 | - |
| **PD** | 𝑟𝑚 𝑡𝑑  𝐾𝑡 (1.25 + 6𝑟 )  𝑑 𝑚 | - | 6 − 2𝑡𝑑  𝑟𝑚  𝑡𝑑 ( 3𝑡𝑑)  22 + 𝑟𝑚 |
| **PID** | 𝑟𝑚 𝑡𝑑  𝐾𝑡 (1.33 + 4𝑟 )  𝑑 𝑚 | 32 + 6𝑡𝑑  𝑟𝑚  𝑡𝑑 ( 8𝑡𝑑)  13 + 𝑟𝑚 | 4  𝑡𝑑 ( 2𝑡𝑑)  11 + 𝑟𝑚 |

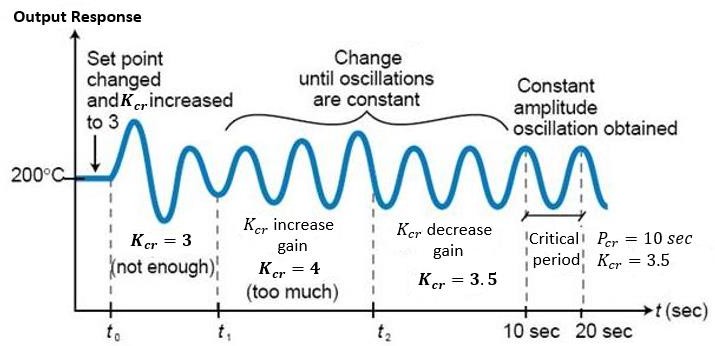
1. **Ziegler-Nichols Closed - Loop Tuning Method**

The Ziegler-Nichols closed-loop tuning method allows you to use the critical gain value, 𝐾𝑐𝑟, and the critical period of oscillation, 𝑃𝑐𝑟, to calculate . It is a simple method of tuning PID controllers and can be refined to give better approximations of the controller. You can obtain the controller constants , 𝑇𝑖 , and 𝑇𝑑 in a system with feedback. The Ziegler-Nichols closed-loop tuning method is limited to tuning processes that cannot run in an open-loop environment.

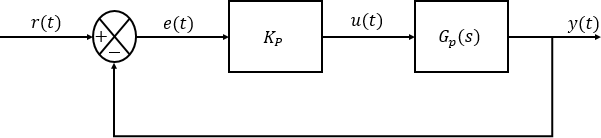
Determining the ultimate gain value, 𝐾𝑐𝑟. is accomplished by finding the value of the proportional- only gain that causes the control loop to oscillate indefinitely at steady state. This means that the gains from the I and D controller are set to zero so that the influence of P can be determined. It tests the robustness of the 𝐾𝑝 value so that it is optimized for the controller. Another important value associated with this proportional-only control tuning method is the critical period (𝑃𝑐𝑟). The ultimate period is the time required to complete one full oscillation while the system is at steady state. These two parameters, 𝐾𝑐𝑟 and 𝑃𝑐𝑟, are used to find the loop-tuning constants of the controller (P, PI, or PID). To find the values of these parameters, and to calculate the tuning constants, use the following procedure:

* The Tuning Procedure:

1. Remove integral and derivative action. Set integral time (𝑇𝑖) to ∞ or its largest value and set the derivative controller (𝑇𝑑) to zero.
2. Create a small disturbance in the loop by changing the set point. Adjust the proportional, increasing and/or decreasing, the gain until the oscillations have constant amplitude.
3. Record the gain value (𝐾𝑐𝑟) and period of oscillation (𝑃𝑐𝑟).
4. Plug these values into the Ziegler-Nichols closed loop equations and determine the necessary settings for the controller.



**Figure 26**: System tuned using the Ziegler-Nichols closed-loop tuning method



**T****able 5**: Closed-Loop Calculation of (𝐾𝑝. 𝑇𝑖. 𝑇𝑑)

|  |  |  |  |
| --- | --- | --- | --- |
|  | 𝑲𝒑 | 𝑻𝒊 | 𝑻𝒅 |
| **P- Controller** | 𝐾𝑐𝑟 2 | ∞ | 0 |
| **PI- Controller** | 𝐾𝑐𝑟 2.2 | 𝑃𝑐𝑟 1.2 | 0 |
| **PID- Controller** | 𝐾𝑐𝑟 1.7 | 𝑃𝑐𝑟 2 | 𝑃𝑐𝑟 8 |

Thus the PID Controllers adds a pole at origin and double zeros at 𝑠 = − 4

𝑃𝑐𝑟

If the system has a known mathematical model (Transfer function is given), then RL method can be used to find *K cr* value (critical gain) and the frequency of the sustained oscillations *w cr* . After that *Pcr* is found from

*Pcr*

 2

*w*

*cr*

These values can be found from the crossing points of the root locus branches with the *jw* axis. This method doesn’t apply if the root locus doesn’t cross the *jw* axis.

* Advantages Ziegler-Nichols Closed-Loop Tuning Methods
  + 1. Easy experiment; only need to change the P controller
    2. Includes dynamics of whole process, which gives a more accurate picture of how the system is behaving
* Disadvantages Ziegler-Nichols Closed-Loop Tuning Methods

1. Experiment can be time consuming
2. Can venture into unstable regions while testing the P controller, which could cause the system to become out of control

# Software Method (PID Tuning Toolbox In MATLAB)

### Automatically Tune PID Controller Gains

PID tuning is the process of finding the values of proportional, integral, and derivative gains of a PID controller to achieve desired performance and meet design requirements.

PID controller tuning appears easy, but finding the set of gains that ensures the best performance of your control system is a complex task. Traditionally, PID controllers are tuned either manually or using rule-based methods. Manual tuning methods are iterative and time-consuming, and if used on hardware, they can cause damage. Rule-based methods also have serious limitations: they do

not support certain types of plant models, such as unstable plants, high-order plants, or plants with little or no time delay.

You can automatically tune PID controllers to achieve the optimal system design and to meet design requirements, even for plant models that traditional rule-based methods cannot handle well.

An automated PID tuning workflow involves:

* Identifying plant model from input-output test data
* Modeling PID controllers in MATLAB using PID objects or in Simulink using PID Controller blocks
* Automatically tuning PID controller gains and fine-tune your design interactively
* Tuning multiple controllers in batch mode
* Tuning single-input single-output PID controllers as well as multiloop PID controller architectures

### PID Tuning Toolbox

Can be use the PID tuning toolbox to determine the parameter of controller depend on the system form MATLAB or SIMULINK as following step:

1. **MATLAB**

http://www.mathworks.com/help/releases/R2015a/control/getstart/pidtune_basic_loop.pngUse the PID Tuner to interactively design a SISO PID controller in the feed-forward path of single-loop, unity-feedback control configuration

The PID Tuner automatically designs a controller for your plant. You specify the controller type (P, I, PI, PD, PDF, PID, PIDF) and form (parallel or standard). You can analyze the design using a variety of response plots, and interactively adjust the design to meet your performance requirements.

To launch the PID Tuner, use the [pidTuner](http://www.mathworks.com/help/control/ref/pidtuner.html) command:

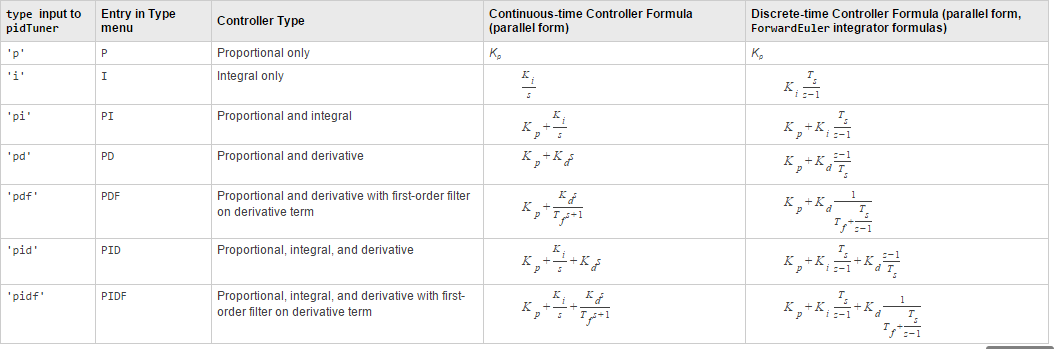
*pidTuner(sys,type)*

where ***sys*** is a linear model of the plant you want to control,

***type*** is a string indicating the [controller type](http://www.mathworks.com/help/control/getstart/designing-pid-controllers-with-the-pid-tuner-gui.html#bsorshv-1) to design

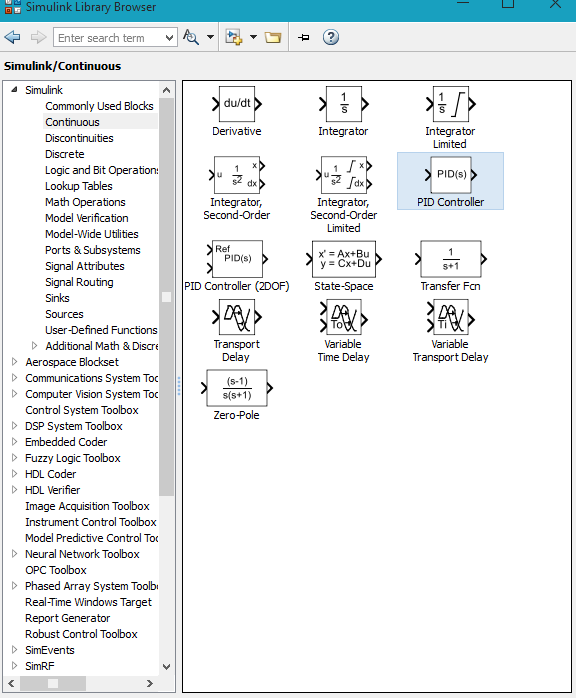
#### PID Controller Type

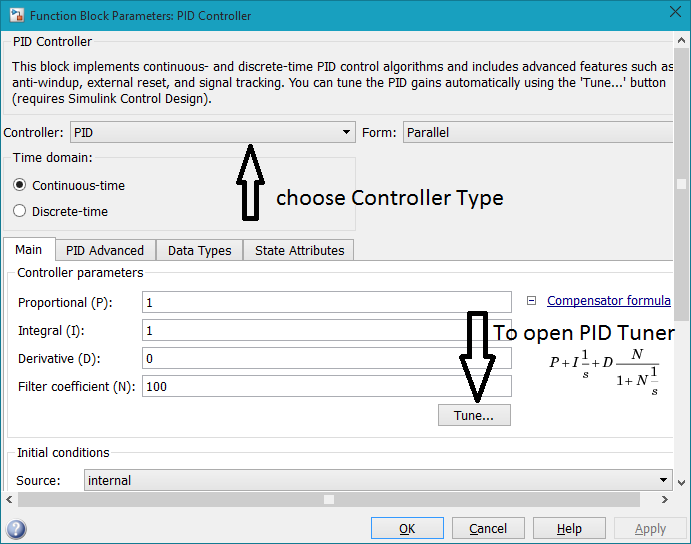
The PID Tuner can tune up to seven types of controllers. To select the controller type, use one of these methods:

Provide the type argument to the launch command pidTuner. In PID Tuner, use the Type menu to change controller types.

#### SIMULINK

Select the PID controller block form Simulink Library



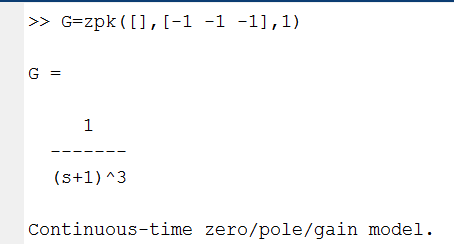
Drag the PID controller and place in the Simulink model , and double click on block.

#### Example 1:

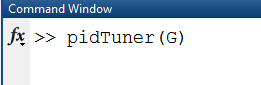
This example shows how to use the PID tuner to design a controller for the plant

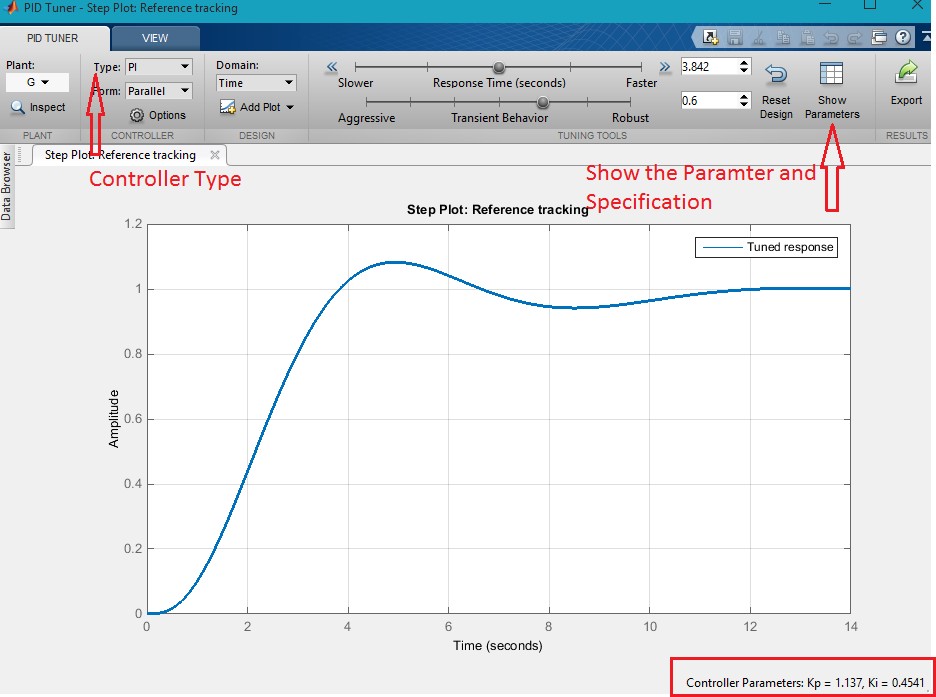
1

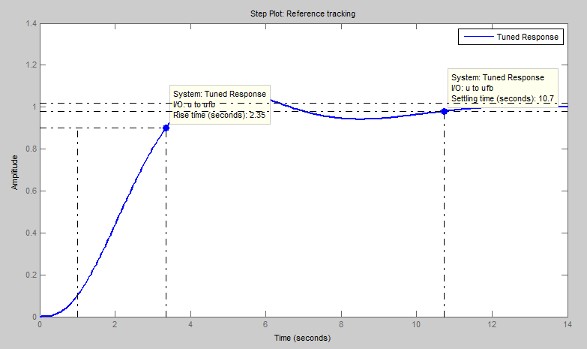
(𝑠) = (𝑠 + 1)3

1. Create the plant model and open the PID Tuner to design a PI controller for a first pass design.

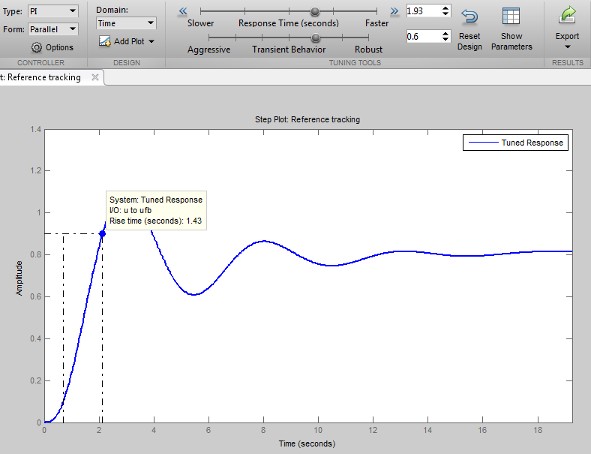
2.

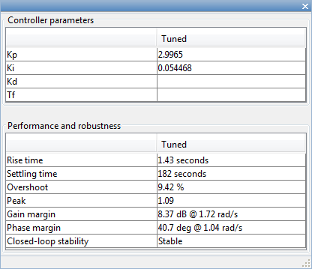


1. The PIDTuner toolbox
2. Examine the reference tracking rise time and settling time.

Right-click on the plot and select ***Characteristics > Rise Time*** to mark the rise time as a blue dot on the plot. Select ***Characteristics > Settling Time to mark the settling time***. To see tool- tips with numerical values, click each of the blue dots.

1. Slide the ***Response time slider*** to the right to try to improve the loop performance. The response plot automatically updates with the new design

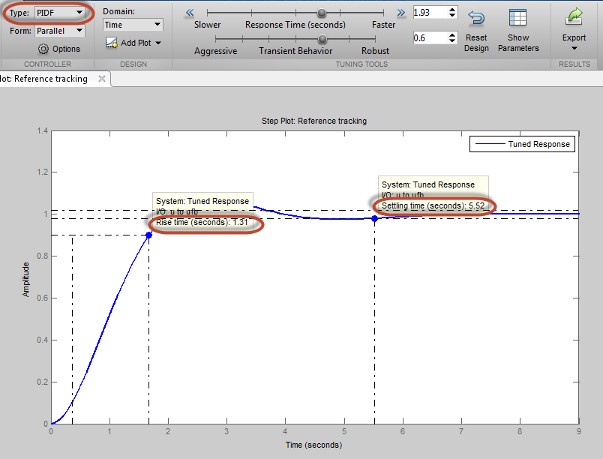


Moving the Response time slider far enough to meet the rise time requirement of less than 1.5 s results in more oscillation. Additionally, the parameters display shows that the new response has an unacceptably long settling time.

To achieve the faster response speed, the algorithm must sacrifice stability.

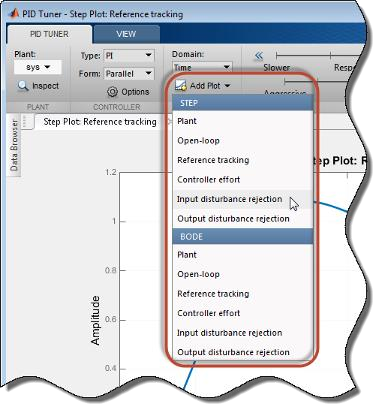
1. Change the controller type to improve the response.

Adding derivative action to the controller gives the PID Tuner more freedom to achieve adequate phase margin with the desired response speed.

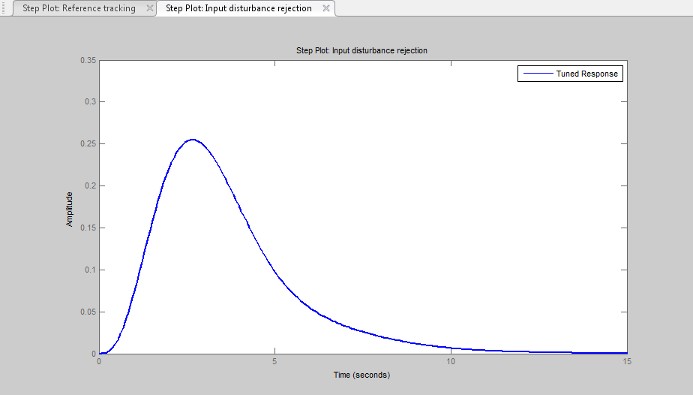
In the Type menu, select PIDF. The PID Tuner designs a new PIDF controller

The rise time and settling time now meet the design requirements. You can use the Response time slider to make further adjustments to the response. To revert to the default automated tuning result, click Reset Design.

1. Analyze other system responses, if appropriate.

To analyze other system responses, click Add Plot. Select the system response you want to analyze.

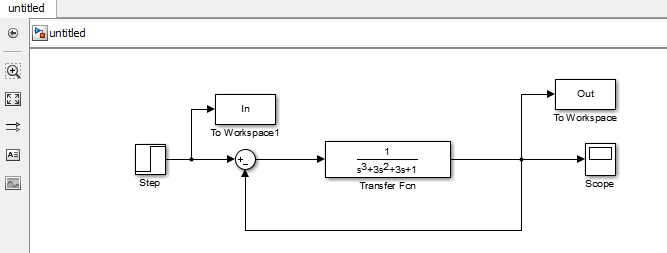
For example, to observe the closed-loop step response to disturbance at the plant input, in the Step section of the **Add Plot** menu, select Input disturbance rejection. The disturbance rejection response appears in a new figure.

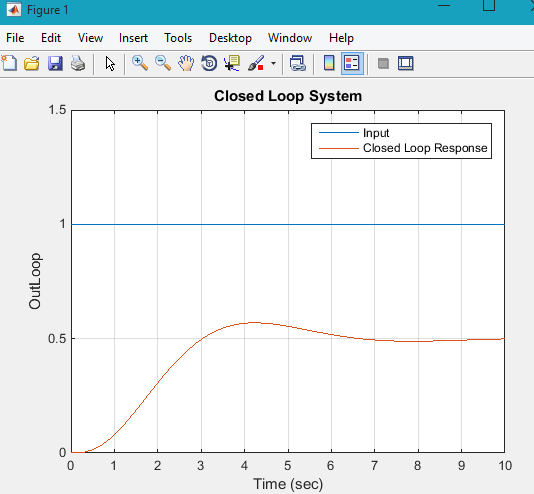


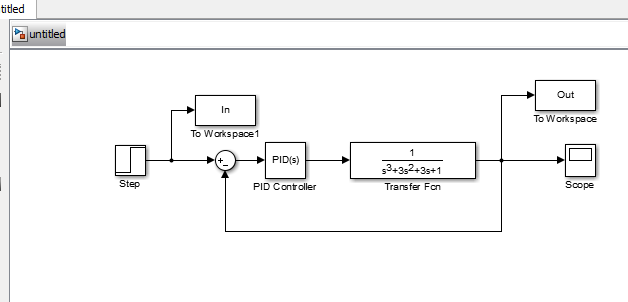
#### Example 2:

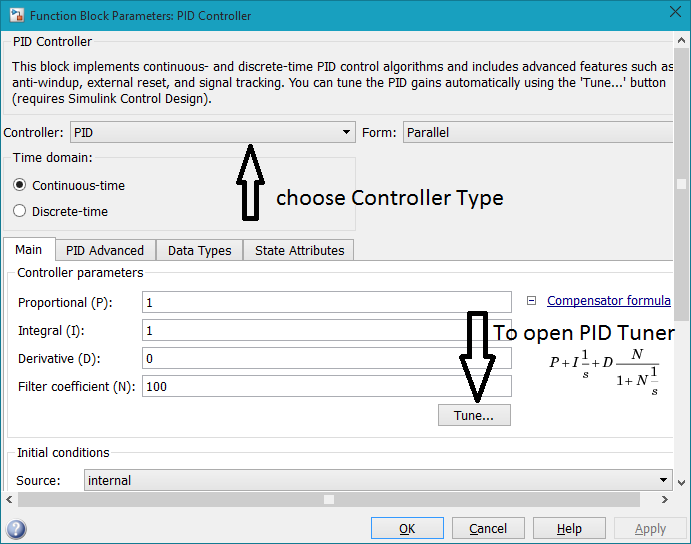
Assume the system in example 1 Bu

* 1. Build the Simulink block represent the closed loop system



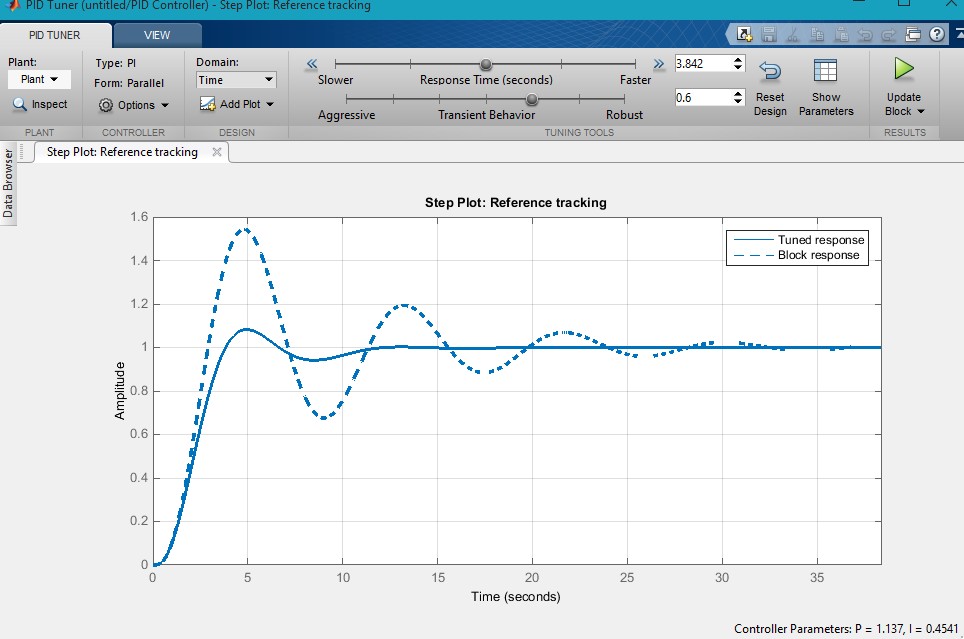


* 1. Add PID Controller block
  2. Double click on PID controller block



* 1. Choose the controller type , click apply and click tune

The PID tuner window will appear and can follow the steps as example 1



#### Note:

If need change the controller type , must be close the tuner window, and repeat step 4

**Prototype implementation**

A gear motor is a combination of gearbox and motor. gear head addition to a motor increases the torque but reduces the speed. To choose suitable DC motor for your application you must calculate loads, speed and torque for your application.

### Motor selection process

#### Design Requirements:

In this phase, the function of the motor is defined and design parameters.

#### Design calculations:

Determination which gets the best chance for the application. Design calculations determine torque, gear ratio, rotating mass, service factor, overhung load, and testing analysis.

#### Motor specifications:

After performing design calculations and defining design parameters, you can use this data to choose the suitable motor for the application. Motor selection is made by some specs:

Power, Voltage, Current, Torque, RPM, Duty Cycle, Rotation (CW or CCW), Shaft Diameter, and Length

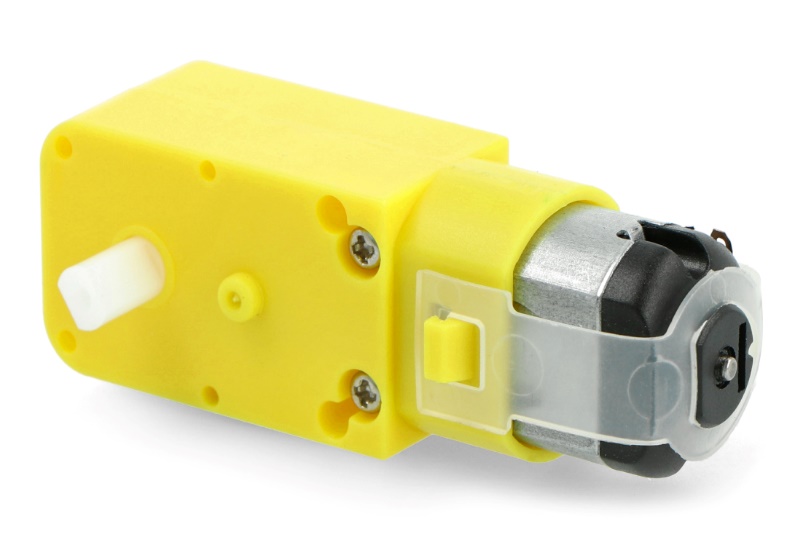
In this project, two types of motors are used according to the application.

### Small DC geared motor:

It is used for prototype to validate the concept.

##### Specification

* Voltage: 5 V
* Current: 180 mA
* Integrated gearbox: 48:1
* Rotation speed after gearbox: 80 rpm.
* Torque after gearbox: 0.5 kg.cm (0.049 Nm)



## Figure 27 dc motor used in prototype

**H-Bridge**

The motor driver (H-Bridge) used to control the speed of that motor is “L298n” which isn’t the most correct motor driver to handle that motor. As the suitable motor driver suitable for this motor is “BTS7960”, but we didn’t find it in the local market and the shipping time was too long, so we used the “L298n” instead of it and it worked properly as we done some modification on it to make it suitable for our goal.

Figure 1 L298n H-bridge

Figure 2 windshield wiper motor

### Theory of operation:

DC Motor: A standard DC motor consists of a rotor (the rotating part) and a stator (the stationary part) connected to a power supply. When voltage is applied to the motor, it spins at a speed proportional to the applied voltage.

Feedback Potentiometer: To convert the DC motor into a servo motor, a potentiometer is attached to the motor shaft. A potentiometer is a three-terminal variable resistor with a movable wiper that slides along a resistive element, changing the resistance between the wiper and the other terminals. This setup allows the potentiometer to measure the motor's angular position.

Control System: A control system is used to compare the desired position (setpoint) of the motor with its actual position (feedback from the potentiometer). It generates an error signal based on the difference between the setpoint and the actual position.

PID Controller: The error signal is then processed by a PID (Proportional-Integral-Derivative) controller. The PID controller calculates a control signal based on the error signal's magnitude and rate of change. The control signal determines the power supplied to the motor.

A diagram of a circuit board

Description automatically generated

**Specifications :**

Power Supply: DC 5 V - 35 V

Peak current: 2 Amp

Operating current range: 0 ~ 36mA

Control signal input voltage range :

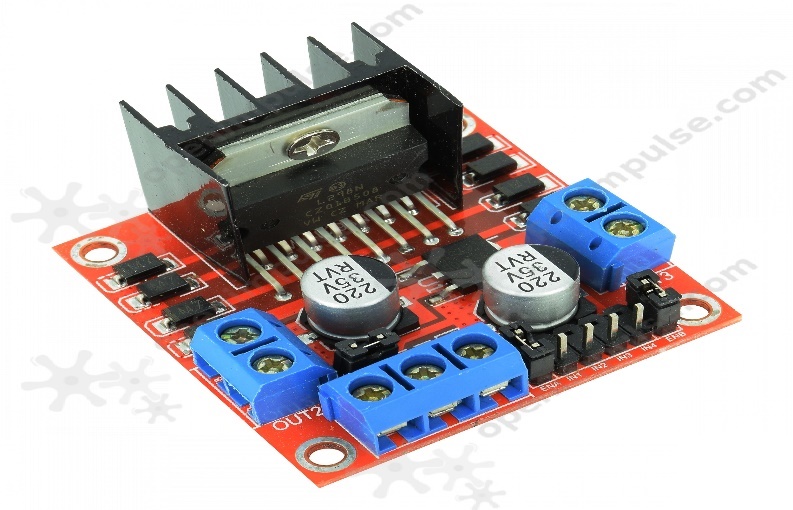
Low: 0V ≤ Vin ≤ 1.5V.

High: 2.3V ≤ Vin ≤ Vss.

Enable signal input voltage range :

Low: 0 ≤ Vin ≤ 1.5V (control signal is invalid).

High: 2.3V ≤ Vin ≤ Vss (control signal active).



### Ultrasonic Specifications



## Figure 28 ultrasonic

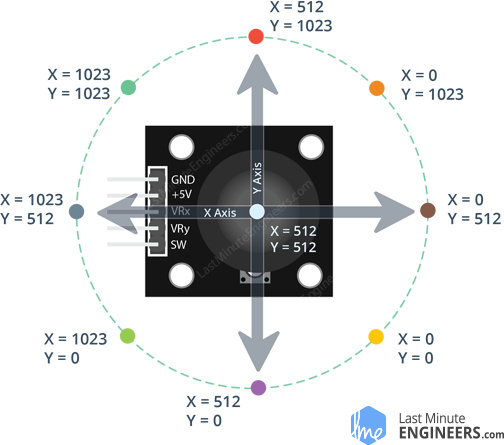
|  |  |
| --- | --- |
| Electrical Parameters | Value |
| Operating Voltage | 3.3Vdc ~ 5Vdc |
| Quiescent Current | <2mA |
| Operating Current | 15mA |
| Operating Frequency | 40KHz |
| Operating Range & Accuracy | 2cm ~ 400cm ( 1in ~ 13ft) ± 3mm |
| Sensitivity | -65dB min |
| Sound Pressure | 112dB |
| Effective Angle | 15° |
| Connector | 4-pin header with 2.54mm pitch |
| Dimension | 45mm x 20mm x 15mm |
| Weight | 9g |

### Table 6: Ultrasonic Specifications

**Joystick**

**Specifications:**

* Operating Voltage: 3V to 5V DC.
* Output Type: The output in the form of analog voltage
* Two independent Potentiometer: one for each axis ( X and Y)
* Internal Potentiometer value: 10k.
* Operating temperature: It operates on the 0 to 70 °C temperature range.

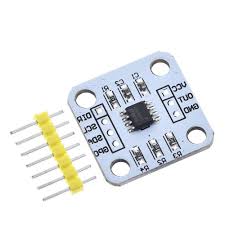
****

## Figure 29 joystick

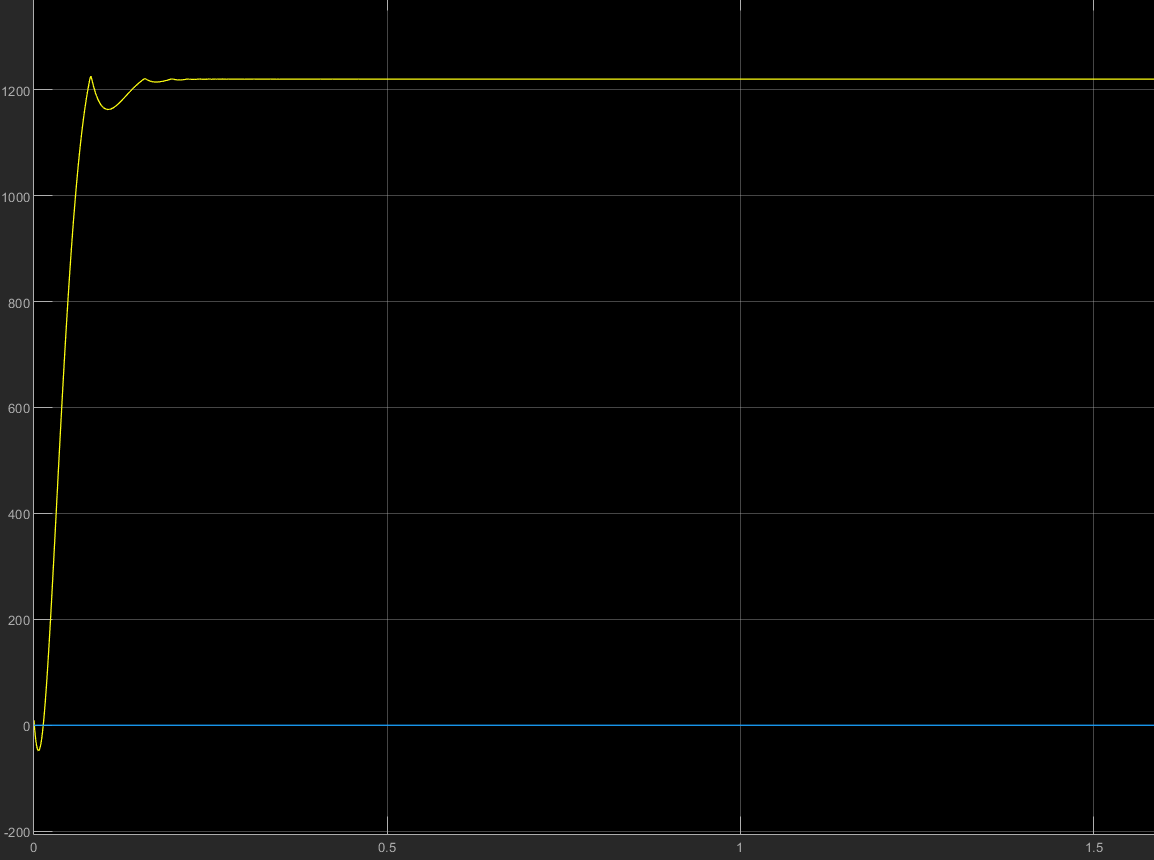
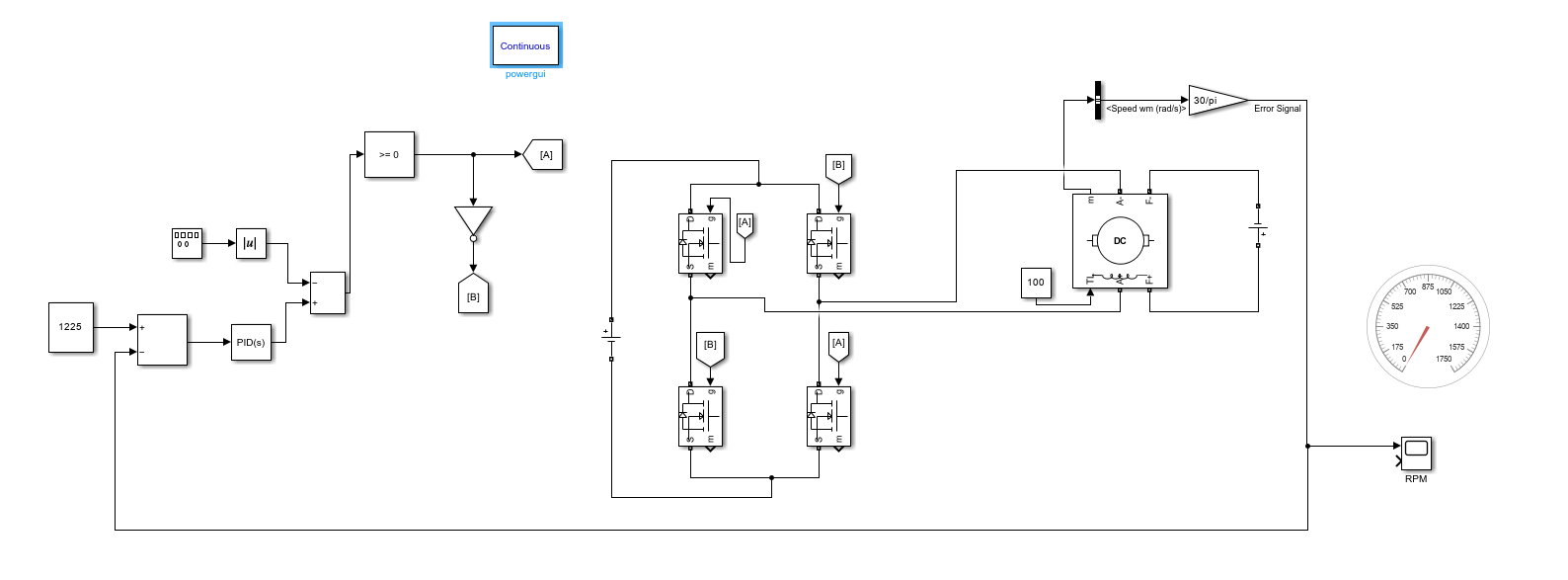
**Magnetic Encoders(AS5600)**

**Specifications:**

* Non-contact, no rotation angle limitation
* 12-bit high-resolution with 4096 positions per round
* Grove I2C and PWM/analog output
* Maximum angle programmable from 18° up to 360°



**Figure** 29 **Magnetic Encoders(AS5600)**

Simulink model 

**flow chart Figure**

**Flow chart**

## Figure 30 flowchart

Int speed = 0

if fwd. state == low

yes

Move fwd. = speed

No

If bw state == low

Move bw = speed

yes

No

If acc == low

yes

Speed +=5

No

If dec == low

yes

No

Speed -=5

If brake == low

yes

No

Speed = 0

**4.4.1 Voice Recognition Integration**

**K10 Wireless Microphone: Overview**

The K10 Wireless Microphone is a sophisticated, portable wireless microphone designed to deliver superior audio quality and user convenience. Featuring advanced noise cancellation technology, it effectively reduces ambient noise, ensuring clear audio input, which is crucial for accurate voice command recognition in voice-controlled wheeling chair projects.

With Bluetooth 5.0 connectivity, the K10 Wireless Microphone offers flexible integration into various system configurations. Its robust battery life provides extended usage, reducing the need for frequent recharging and ensuring continuous operation. The ergonomic design ensures comfort during extended use,

**Arduino code**

#define in1 2

#define in2 3

#define in3 4

#define in4 5

#define enA 9

#define enB 10

int motorSpeed =65 ;

const int triggerPin =20;

const int echoPin = 21;

const int buzz = 52;

int  xAxis, yAxis;

long duration;

int dist;

int SW\_pin = 14; // Change this to the pin number where your button is connected

int buttonState = 0;

int lastButtonState = 0;

int counter = 0;

void forward(){

    digitalWrite(in1, LOW);

    digitalWrite(in2, HIGH);

    digitalWrite(in3, HIGH);

    digitalWrite(in4, LOW);

    analogWrite(enA, motorSpeed);

    analogWrite(enB, motorSpeed);

}

void backward(){

    digitalWrite(in1, HIGH);

    digitalWrite(in2, LOW);

    digitalWrite(in3, LOW);

    digitalWrite(in4, HIGH);

    analogWrite(enA, motorSpeed);

    analogWrite(enB, motorSpeed);

}

void left(){

    digitalWrite(in1, LOW);

    digitalWrite(in2, LOW);

    digitalWrite(in3, HIGH);

    digitalWrite(in4, LOW);

    analogWrite(enA, motorSpeed);

    analogWrite(enB, motorSpeed);

}

void right(){

    digitalWrite(in1, LOW);

    digitalWrite(in2, HIGH);

    digitalWrite(in3, LOW);

    digitalWrite(in4, LOW);

    analogWrite(enA, motorSpeed);

    analogWrite(enB, motorSpeed);

}

void stop(){

    digitalWrite(in1, LOW);

    digitalWrite(in2, LOW);

    digitalWrite(in3, LOW);

    digitalWrite(in4, LOW);

    analogWrite(enA, 0);

    analogWrite(enB, 0);

}

void ultrasonic(){

  digitalWrite(triggerPin, LOW);

  delayMicroseconds(2);

  digitalWrite(triggerPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(triggerPin, LOW);

  duration = pulseIn(echoPin, HIGH);

  dist = duration \* 0.034 / 2;

  Serial.print("Distance: ");

  Serial.println(dist);

  if (dist < 200) {

    digitalWrite(buzz,HIGH);

  for (int i=dist; i>0; i--)

    delay(1);

    for (int i=dist; i>0; i--)

    delay(1);

  digitalWrite(buzz,LOW);

  for (int i=dist; i>0; i--)

    delay(1);

    for (int i=dist; i>0; i--)

    delay(1);   }

}

void setup() {

  Serial.begin(9600);

  pinMode(SW\_pin, INPUT\_PULLUP);

  pinMode(enA, OUTPUT);

  pinMode(enB, OUTPUT);

  pinMode(in1, OUTPUT);

  pinMode(in2, OUTPUT);

  pinMode(in3, OUTPUT);

  pinMode(in4, OUTPUT);

  pinMode(triggerPin, OUTPUT);

  pinMode(echoPin, INPUT);

  pinMode(buzz, OUTPUT);

}

void loop() {

  ultrasonic();

  buttonState = digitalRead(SW\_pin);

  if (lastButtonState == HIGH && buttonState == LOW) {

    counter++;

    motorSpeed+=45;

    Serial.print(motorSpeed);

  }

  lastButtonState = buttonState;

  delay(20);

  if (counter > 4) {

    Serial.println("Button pressed 3 times, resetting counter.");

    counter = 0;

    motorSpeed=70;

    Serial.print(motorSpeed);

  }

  xAxis = analogRead(A0); // Read Joysticks X-axis

  yAxis = analogRead(A1); // Read Joysticks Y-axis

  Serial.println(xAxis);

  Serial.println(yAxis);

  Serial.println(motorSpeed);

  // Y-axis used for forward and backward control

  if (0 <= xAxis <= 1023 && yAxis > 1000  ) {   // Testing for mapping

    // Set Motor backward

   backward();

  }

  else if (0 <= xAxis <= 1023 && yAxis < 20 ) {

    // Set Motor forward

   forward();

  }

  else if(xAxis <20 && 0 <= yAxis <= 1023 ) {

    right();

  }

  else if ( xAxis > 1000 && 0 <= yAxis <= 1023 ) {

 left();

  }

  else {

  stop();

}

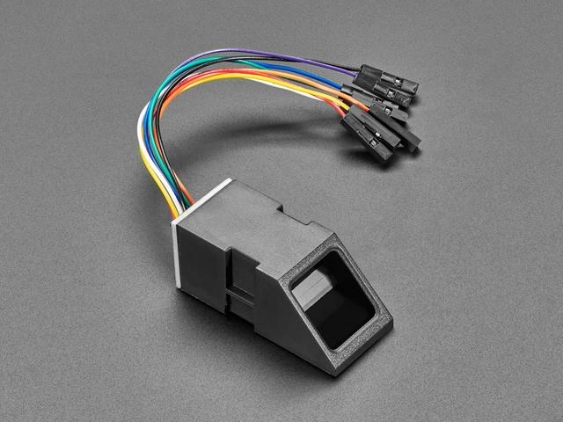
while its compact size makes it less obtrusive compared to over-ear headphones.

High-resolution audio certification guarantees rich and clear sound, enhancing the accuracy of voice recognition. While the K10 Wireless Microphone's design is more convenient and portable compared to over-ear headphones, it may face challenges such as occasional Bluetooth interference and higher costs compared to simpler devices.

**FINGERPRINT (AS608)**

Secure your project with biometrics - this all-in-one optical fingerprint sensor will make adding fingerprint detection and verification super simple. These modules are typically used in safes - there's a high powered DSP chip that does the image rendering, calculation, feature-finding and searching. Connect to any microcontroller or system with TTL serial, and send packets of data to take photos, detect prints, hash and search. You can also enroll new fingers directly - up to 162 finger prints can be stored in the onboard FLASH memory.

**AS608 Characteristics**



Supply voltage: 3.6 - 6.0VDC

Operating current: 120mA max

Peak current: 150mA max

Fingerprint imaging time: <1.0 seconds

Window area: 14mm x 18mm

Signature file: 256 bytes

Template file: 512 bytes

Storage capacity: 162 templates

Safety ratings (1-5 low to high safety)

False Acceptance Rate: <0.001% (Security level 3)

False Reject Rate: <1.0% (Security level 3)

Interface: TTL Serial

Baud rate: 9600, 19200, 28800, 38400, 57600 (default is 57600)

Working temperature rating: -20C to +50C

Working humidy: 40%-85% RH

Full Dimensions: 56 x 20 x 21.5mm

Exposed Dimensions (when placed in box): 21mm x 21mm x 21mm triangular

Weight: 20 grams

**Way of use**

There are basically two requirements for using the optical fingerprint sensor. First is you'll need to enroll fingerprints - that means assigning ID #'s to each print so you can query them later. Once you've enrolled all your prints, you can easily 'search' the sensor, asking it to identify which ID (if any) is currently being photographed.

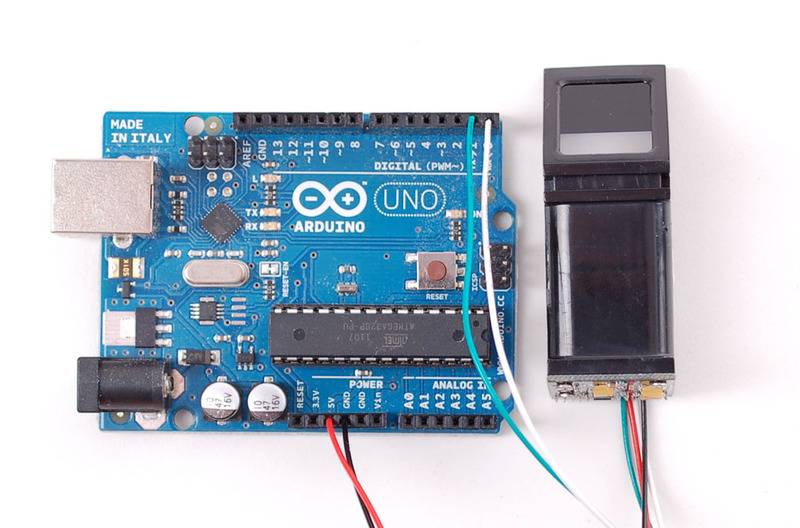
You can enroll using the Windows software (easiest and neat because it shows you the photograph of the print) or with the Arduino sketch (good for when you don't have a Windows machine handy or for on-the-road enrolling)

**Python library installation**

You'll need to install the Adafruit\_Blinka library that provides the CircuitPython support in Python. This may also require enabling the hardware UART on your platform (see red note above) and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready (https://adafru.it/BSN)!

Once that's done, from your command line run the following command: sudo pip3 install adafruit-circuitpython-fingerprint

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!



**Code**

# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries

# SPDX-License-Identifier: MIT

import time

import board

import busio

from digitalio import DigitalInOut, Direction

import adafruit\_fingerprint

led = DigitalInOut(board.D13)

led.direction = Direction.OUTPUT

uart = busio.UART(board.TX, board.RX, baudrate=57600)

# If using with a computer such as Linux/RaspberryPi, Mac, Windows with USB/serial

converter:

# import serial

# uart = serial.Serial("/dev/ttyUSB0", baudrate=57600, timeout=1)

# If using with Linux/Raspberry Pi and hardware UART:

# import serial

# uart = serial.Serial("/dev/ttyS0", baudrate=57600, timeout=1)

finger = adafruit\_fingerprint.Adafruit\_Fingerprint(uart)

##################################################

def get\_fingerprint():

"""Get a finger print image, template it, and see if it matches!"""

print("Waiting for image...")

while finger.get\_image() != adafruit\_fingerprint.OK:

pass

print("Templating...")

if finger.image\_2\_tz(1) != adafruit\_fingerprint.OK:

return False

print("Searching...")

if finger.finger\_search() != adafruit\_fingerprint.OK:

return False

return True

# pylint: disable=too-many-branches

def get\_fingerprint\_detail():

"""Get a finger print image, template it, and see if it matches!

This time, print out each error instead of just returning on failure"""

print("Getting image...", end="")

i = finger.get\_image()

if i == adafruit\_fingerprint.OK:

print("Image taken")else:

if i == adafruit\_fingerprint.NOFINGER:

print("No finger detected")

elif i == adafruit\_fingerprint.IMAGEFAIL:

print("Imaging error")

else:

print("Other error")

return False

print("Templating...", end="")

i = finger.image\_2\_tz(1)

if i == adafruit\_fingerprint.OK:

print("Templated")

else:

if i == adafruit\_fingerprint.IMAGEMESS:

print("Image too messy")

elif i == adafruit\_fingerprint.FEATUREFAIL:

print("Could not identify features")

elif i == adafruit\_fingerprint.INVALIDIMAGE:

print("Image invalid")

else:

print("Other error")

return False

print("Searching...", end="")

i = finger.finger\_fast\_search()

# pylint: disable=no-else-return

# This block needs to be refactored when it can be tested.

if i == adafruit\_fingerprint.OK:

print("Found fingerprint!")

return True

else:

if i == adafruit\_fingerprint.NOTFOUND:

print("No match found")

else:

print("Other error")

return False

# pylint: disable=too-many-statements

def enroll\_finger(location):

"""Take a 2 finger images and template it, then store in 'location'"""

for fingerimg in range(1, 3):

if fingerimg == 1:

print("Place finger on sensor...", end="")

else:

print("Place same finger again...", end="")

while True:

i = finger.get\_image()

if i == adafruit\_fingerprint.OK:

print("Image taken")

break

if i == adafruit\_fingerprint.NOFINGER:

print(".", end="")

elif i == adafruit\_fingerprint.IMAGEFAIL:

print("Imaging error")

return False

else:

print("Other error")

return False

print("Templating...", end="")

i = finger.image\_2\_tz(fingerimg)

if i == adafruit\_fingerprint.OK:

print("Templated")

else:

if i == adafruit\_fingerprint.IMAGEMESS:

print("Image too messy")elif i == adafruit\_fingerprint.FEATUREFAIL:

print("Could not identify features")

elif i == adafruit\_fingerprint.INVALIDIMAGE:

print("Image invalid")

else:

print("Other error")

return False

if fingerimg == 1:

print("Remove finger")

time.sleep(1)

while i != adafruit\_fingerprint.NOFINGER:

i = finger.get\_image()

print("Creating model...", end="")

i = finger.create\_model()

if i == adafruit\_fingerprint.OK:

print("Created")

else:

if i == adafruit\_fingerprint.ENROLLMISMATCH:

print("Prints did not match")

else:

print("Other error")

return False

print("Storing model #%d..." % location, end="")

i = finger.store\_model(location)

if i == adafruit\_fingerprint.OK:

print("Stored")

else:

if i == adafruit\_fingerprint.BADLOCATION:

print("Bad storage location")

elif i == adafruit\_fingerprint.FLASHERR:

print("Flash storage error")

else:

print("Other error")

return False

return True

##################################################

def get\_num():

"""Use input() to get a valid number from 1 to 127. Retry till success!"""

i = 0

while (i > 127) or (i < 1):

try:

i = int(input("Enter ID # from 1-127: "))

except ValueError:

pass

return i

while True:

print("----------------")

if finger.read\_templates() != adafruit\_fingerprint.OK:

raise RuntimeError("Failed to read templates")

print("Fingerprint templates:", finger.templates)

print("e) enroll print")

print("f) find print")

print("d) delete print")

print("----------------")

c = input("> ")

if c == "e":

enroll\_finger(get\_num())

if c == "f":if get\_fingerprint():

print("Detected #", finger.finger\_id, "with confidence",

finger.confidence)

else:

print("Finger not found")

if c == "d":

if finger.delete\_model(get\_num()) == adafruit\_fingerprint.OK:

print("Deleted!")

else:

print("Failed to delete")

**What are the procces taken from start of taking the voice until camera and control:**

The process of converting voice commands into actions (such as camera control and motor movement) in a voice-controlled wheelchair involves several key steps, from capturing the user's voice to executing the appropriate actions. Here is a detailed explanation of the process:

#### 1. Voice Capture

**Microphone Input:**

* The process begins with the user speaking a command.
* A high-sensitivity microphone captures the user's voice.
* The audio signal is then converted into a digital format using an analog-to-digital converter (ADC).

#### 2. Voice Recognition

**Preprocessing:**

* The captured digital audio signal is preprocessed to remove noise and enhance the quality of the speech signal.
* Techniques such as filtering, normalization, and feature extraction (e.g., Mel-Frequency Cepstral Coefficients - MFCC) are used.

**Speech Recognition Engine:**

* The preprocessed audio is fed into a speech recognition engine, such as Deep Speech or VOSK.
* The engine uses deep learning models to analyze the audio and convert it into a string of text representing the spoken command.

**Example:**

* User says: "Move forward"
* Recognized text: "move forward"

#### 3. Command Parsing

**Natural Language Processing (NLP):**

* The recognized text is processed using NLP techniques to understand the intent of the command.
* Command parsing involves breaking down the text into actionable parts.

**Example:**

* Command: "move forward"
* Parsed command: {action: "move", direction: "forward"}

#### 4. Command Interpretation and Action Mapping

**Command Interpreter:**

* The parsed command is sent to a command interpreter.
* The interpreter maps the command to specific actions that the system can execute.

**Action Mapping:**

* Each command corresponds to a predefined set of actions for the motors and camera.
* For example, "move forward" might be mapped to increasing the motor speed in the forward direction.

**Example Mapping:**

* "Move forward" -> Increase motor speed forward
* "Turn left" -> Adjust motor speed for a left turn

#### 5. Control Signal Generation

**Control Algorithms:**

* Control algorithms generate specific signals for the motors and camera based on the interpreted commands.
* For motors, this could involve generating PWM (Pulse Width Modulation) signals to control speed and direction.

**Microcontroller Unit (MCU):**

* The control signals are sent to the microcontroller unit (e.g., Arduino Mega) which interfaces with the motors and other hardware components.

#### 6. Execution by Hardware Components

**Motor Control:**

* The microcontroller sends signals to the motor drivers to control the speed and direction of the wheelchair's motors.
* For example, increasing PWM duty cycle for forward movement or differential speed for turning.

**Camera Control (if applicable):**

* If the command involves camera movement (e.g., "look left"), the microcontroller sends control signals to the camera servo motors to adjust its position.

**Feedback Mechanisms:**

* Encoders on the motors provide feedback to ensure precise control and adjust as necessary.
* Sensors such as distance sensors provide environmental feedback to avoid obstacles.

#### 7. Real-time Adjustments and Safety

**Safety Checks:**

* The system continuously monitors sensors (e.g., proximity sensors) to ensure safe operation.
* If an obstacle is detected, the system can override commands to prevent collisions.

### Example Flow

1. **User Command:** "Move forward"
2. **Voice Capture:** Microphone captures the audio.
3. **Voice Recognition:** Audio is converted to text "move forward."
4. **Command Parsing:** Text is parsed into {action: "move", direction: "forward"}.
5. **Command Interpretation:** Parsed command is mapped to motor action.
6. **Control Signal Generation:** Control signals for forward movement are generated.
7. **Execution:** Microcontroller sends signals to motors to move the wheelchair forward.
8. **Safety Checks:** Sensors ensure there are no obstacles in the path.

**VOSK Speech Recognition**

Using the voice recognition module was difficult because it is required to be in a perfect environment such as the noise in sample records must be the same in the verification record. So, we decided to use a more efficient method to use this feature in real-time.

**4.4.1.1 Why VOSK**

We are using the VOSK module for our voice-controlled wheeling chair project because it provides high accuracy and efficiency in speech recognition while operating entirely offline, ensuring reliability without internet dependence. It supports multiple languages, making it versatile for diverse users, and has low resource requirements, suitable for limited processing power systems. VOSK’s easy integration with Python, scalable modular architecture, and extensive support resources make it an ideal choice for enhancing the chair's functionality and accessibility through voice control.

**4.4.1.2 Model used: VOSK-Model-Small-en-us-0.15**

The VOSK-Model-Small-en-us-0.15 is a compact and efficient speech recognition model optimized for the English language. It is part of the VOSK suite, renowned for their high accuracy and minimal computational requirements. This model is specifically designed for use in applications with limited resources, making it an ideal choice for deployment on devices like the Raspberry Pi. The model uses a dataset of approximately 40 hours of speech data and achieves an accuracy rate of around 90% for general English speech recognition tasks. It is released under the Apache 2.0 License, ensuring it is free to use and distribute.

**4.4.1.3 Rationale for Using VOSK-Model-Small-en-us-0.15**

The decision to employ the VOSK-Model-Small-en-us-0.15 was driven by several key factors:

* Efficiency and Lightweight Design: This model’s small size allows it to operate efficiently on the Raspberry Pi, a device characterized by limited processing power and memory capacity. This ensures that the model can run smoothly without overwhelming the system.
* Offline Functionality: The model’s ability to function offline is crucial for applications where internet connectivity may be intermittent or unavailable. This ensures reliable and uninterrupted operation.
* High Recognition Accuracy: Despite its compact size, the VOSK-Model-Small-en-us-0.15 offers robust speech recognition capabilities, accurately capturing and interpreting voice commands essential for our voice-controlled wheeling chair project.
* License: The model is distributed under the Apache 2.0 License, which allows for free use, modification, and distribution, making it an excellent choice for open-source projects.

**4.4.1.4 Implementation on Raspberry Pi**

The implementation of the VOSK-Model-Small-en-us-0.15 on a Raspberry Pi involves several steps:

1. System Preparation:
   * Ensure the Raspberry Pi is updated with the latest software.
   * Install the necessary dependencies, including Python and the VOSK library.
2. Model Installation:
   * Install the VOSK library using pip: pip install vosk.
   * Download the VOSK-Model-Small-en-us-0.15 and extract it to a designated directory on the Raspberry Pi.
3. Integration with Application:
   * Integrate the VOSK model into the Python application using the VOSK library.
   * Configure the microphone input to capture voice commands, which are then processed by the VOSK model.
   * Implement a queue system to handle recognized commands and transmit them to the Arduino for execution.
4. Testing and Optimization:
   * Conduct thorough testing in various environments to ensure the accuracy and responsiveness of the system.
   * Optimize the placement of the microphone and adjust model parameters as necessary to enhance performance.

**4.4.1.5 Challenges with Larger Models**

The project initially considered using a dataset from Kaggle with the DeepSpeech model. However, this approach proved impractical due to several significant challenges:

* **Resource Limitations**: The Kaggle dataset and DeepSpeech model were too large to be efficiently deployed on the Raspberry Pi. The Raspberry Pi's limited processing power and memory capacity could not handle the extensive computational requirements of these resources.
* **Performance Issues**: Attempting to run the larger DeepSpeech model on the Raspberry Pi led to CPU overload, causing the device to crash and become unstable. Additionally, the sophisticated techniques required by DeepSpeech were beyond the capabilities of the Raspberry Pi, leading to further performance degradation.
* **Internet Dependency**: Initially, the voice recognition system required an internet connection to function. This dependency posed a significant challenge, as reliable internet connectivity could not always be guaranteed. This limitation hindered the system's reliability and practicality in real-world applications.

Due to these constraints, we selected the VOSK-Model-Small-en-us-0.15, which offers a balance between performance and resource efficiency, ensuring reliable operation on the Raspberry Pi. The VOSK model's lightweight design, offline functionality, and high recognition accuracy make it an ideal choice for our voice-controlled wheeling chair project. The ability to operate entirely offline has significantly enhanced the system's reliability and usability, eliminating the need for a constant internet connection.

By employing the VOSK-Model-Small-en-us-0.15, we ensure a reliable and efficient speech recognition system for our voice-controlled wheeling chair project. This model meets the project’s requirements for accurate command recognition and efficient processing within the limitations of the Raspberry Pi, thereby enhancing the functionality and user experience of the wheeling chair.

**4.4.1.6 Evaluation**

The evaluation criteria for the VOSK Speech Recognition Library’s performance include latency, accuracy, and robustness:

* Latency: Measures the library’s response time, ensuring it can process commands quickly.
* Accuracy: Assesses the correct recognition of spoken words, which is crucial for reliable operation.
* Robustness: Evaluates performance with different speakers and varying background noise levels.

Test scenarios were designed to evaluate the library under various conditions, such as different speech patterns, noise levels, and speakers. These tests provide insights into the library’s limitations and strengths.

**4.4.1.7 Challenges**

1. Word Recognition Errors: Certain words may be incorrectly recognized due to accent variations or similar-sounding words. For instance, “backward” could be misinterpreted. Ensuring the use of clearly distinguishable words can mitigate this issue.
2. Hardware Limitations: Running VOSK on a Raspberry Pi or similar hardware might be constrained by processing power. Ensuring the system is optimized and considering hardware upgrades can help maintain performance.
3. Customization Complexity: Customizing the VOSK models for specific applications might require advanced knowledge and technical skills. Providing adequate training and documentation for developers can ease this process.

**4.4.1.8 Code**

**Description of Python Code**

The Python code is a graphical application that uses speech recognition to control the direction of an arrow displayed on the screen. It also communicates with an Arduino via serial communication to send the recognized commands.

**Overview**

1. Speech Recognition: The application uses the Vosk model for speech recognition to process voice commands.
2. GUI: The graphical user interface (GUI) is built using Tkinter.
3. Arduino Communication: The application sends recognized commands to an Arduino via a serial connection.

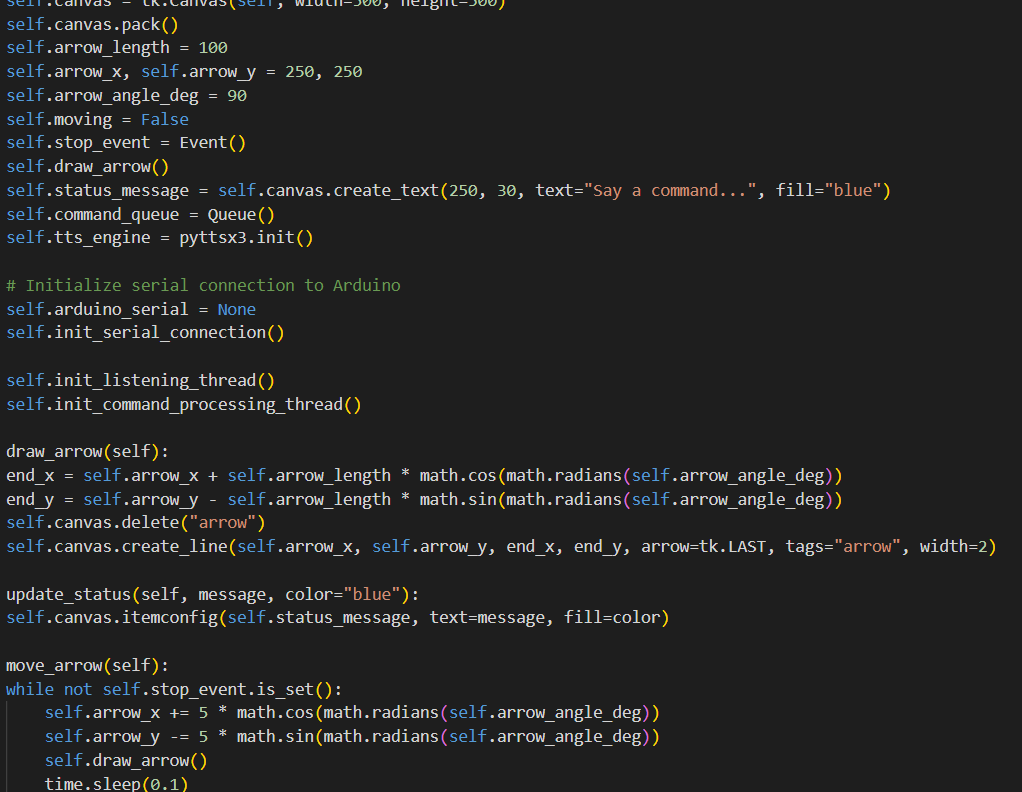
**Components**

1. Import Statements: Import necessary libraries for the application.
   * tkinter: For GUI components.
   * threading: For running tasks concurrently.
   * queue: For managing command tasks.
   * math: For mathematical calculations.
   * speech\_recognition: For capturing and processing audio.
   * json: For handling JSON data.
   * pyttsx3: For text-to-speech functionality.
   * vosk: For speech recognition model.
   * serial: For serial communication.
   * time: For managing delays.
   * difflib: For finding close matches to commands.
2. Speech Recognition Function (run\_vosk\_recognition):
   * Initializes the Vosk model and recognizer.
   * Continuously captures audio input and processes it to recognize speech.
   * Puts recognized commands into a queue for further processing.
3. Main Application Class (VoiceControlledArrow):
   * Initialization (\_\_init\_\_):
     + Sets up the Tkinter window.
     + Initializes the arrow's position and angle.
     + Creates a canvas for drawing the arrow.
     + Initializes the serial connection to the Arduino.
     + Starts the speech recognition and command processing threads.
   * Drawing the Arrow (draw\_arrow): Updates the arrow's position on the canvas.
   * Updating Status (update\_status): Updates the status message on the GUI.
   * Moving the Arrow (move\_arrow): Continuously moves the arrow in the current direction.
   * Processing Commands (update\_arrow\_direction\_and\_move): Updates the arrow's direction and sends commands to the Arduino.
   * Text-to-Speech (speak\_text): Converts text to speech.
   * Serial Connection Initialization (init\_serial\_connection): Establishes the serial connection to the Arduino.
   * Sending Commands to Arduino (send\_command\_to\_arduino): Sends recognized commands to the Arduino.
   * Reading Responses from Arduino (read\_response\_from\_arduino): Reads responses from the Arduino (not actively used in this version).
   * Graceful Shutdown (on\_close): Ensures the serial connection is closed when the application exits.
   * Running the Application (run): Starts the Tkinter main loop.

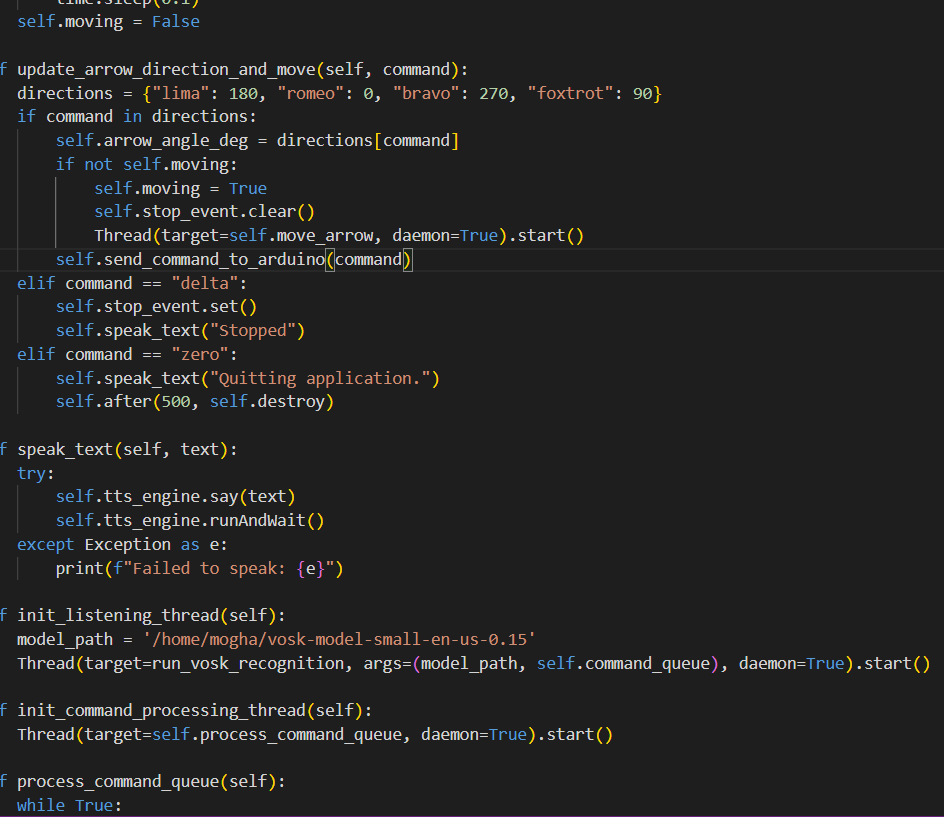
and here is the code :



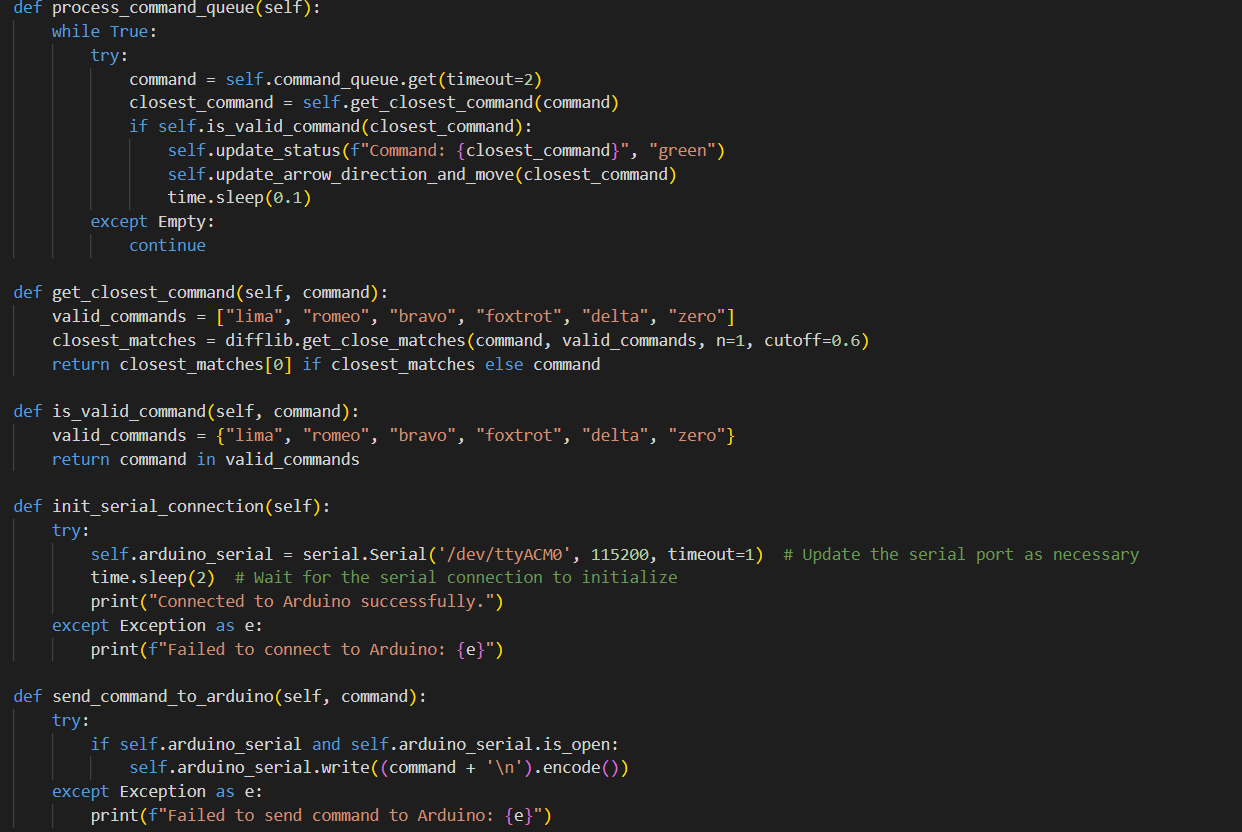
**Fig 4.2 coding the voice command**

****

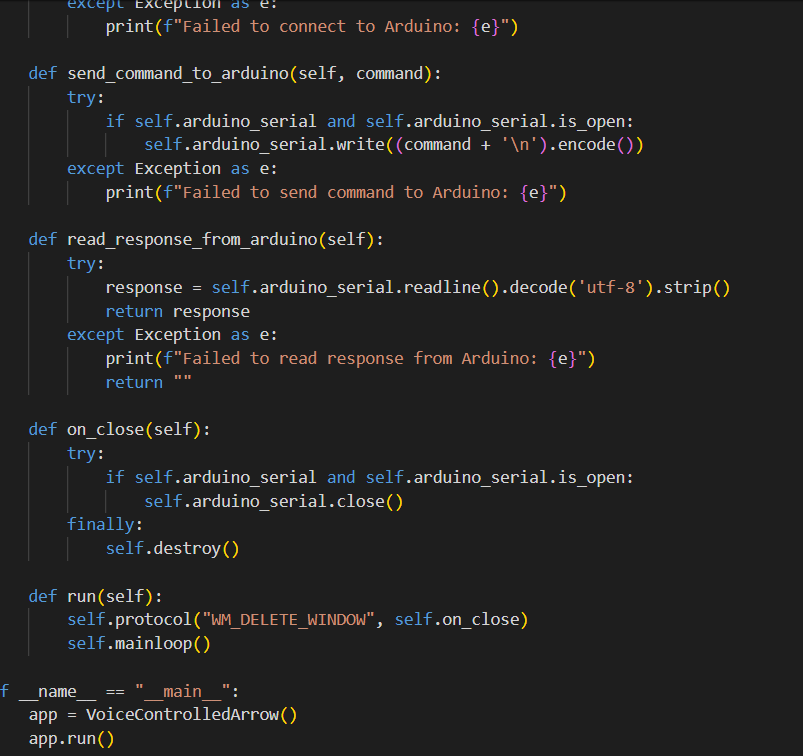
**Fig 4.3 coding the voice command**

****

**Fig 4.4 coding the voice command**

****

**Fig 4.5 coding the voice command**

****

**Fig 4.6 coding the voice command**

for the arduino this code converts the ASCII value of the first letter of the received command into a string and then prints it

setup() Function: This function runs once when the Arduino is powered on or reset.

* Serial.begin(115200): Initializes the serial communication at a baud rate of 115,200 bits per second.
* while (!Serial) { ; }: Waits for the serial port to connect. This is needed for boards with native USB ports (e.g., Arduino Leonardo, Micro).
* Serial.println("Arduino is ready."): Prints the message "Arduino is ready." to the serial monitor.

loop() Function: This function runs continuously in a loop after setup().

* if (Serial.available() > 0): Checks if data is available to read from the serial port.
* String command = Serial.readStringUntil('\n');: Reads the incoming serial data as a string until a newline character (\n) is encountered.
* command.trim();: Removes any leading or trailing whitespace or newline characters from the string.
* if (command.length() > 0): Checks if the command string is not empty.
  + Serial.print("Received command: ");: Prints "Received command: " to the serial monitor.
  + Serial.println(command);: Prints the actual command received to the serial monitor.
  + char firstLetter = command.charAt(0);: Retrieves the first character of the command string.
  + int asciiValue = firstLetter;: Converts the first character to its ASCII value.
  + String asciiString = String(asciiValue);: Converts the ASCII value to a string.
  + Serial.print("First letter: ");: Prints "First letter: " to the serial monitor.
  + Serial.println(firstLetter);: Prints the first character of the command.
  + Serial.print("ASCII string: ");: Prints "ASCII string: " to the serial monitor.
  + Serial.println(asciiString);: Prints the ASCII string representation of the first character.
* else { Serial.println("Empty command received."); }: If the command string is empty, it prints "Empty command received." to the serial monitor.

****

**Fig 4.7 converting the voice command into string**

**4.4.2 Computer Vision Integration**

**4.1.3.3 step by step development approach**

**4.1.3.3.1 Step 1: Train Object Detection Models**

##### **A. Data Collection**

1. Dataset Preparation:
   * Collect images containing humans, tables, and chairs from various angles and lighting conditions.
   * Label the images with bounding boxes around the objects.

##### B. Model Training

1. Choosing a Model:
   * Select a pre-trained object detection model such as YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), or Faster R-CNN.
2. Transfer Learning:
   * Fine-tune the chosen model with your labeled dataset to recognize humans, tables, and chairs.
3. Training Environment:
   * Use a powerful machine with a good GPU (like an NVIDIA GPU) to speed up the training process.
4. Training Framework:
   * Use frameworks such as TensorFlow, PyTorch, or OpenCV for training and implementation.

**4.2.2 computer vision**

Model Evaluation:

* Evaluate the model's performance using metrics such as precision, recall, and mean Average Precision (mAP).

Testing:

* Test the model on unseen images to ensure it generalizes well to new data

**4.2.2.1** **Step 2: Distance Measurement**

##### **A. Depth Estimation**

1. **Stereo Vision:**
   * Use stereo cameras to capture two images from slightly different angles.
   * Compute the disparity map (difference between the two images) to estimate depth.
2. **Monocular Depth Estimation:**
   * Alternatively, use a single camera and a depth estimation algorithm trained to predict depth from monocular images.

##### **B. Implementing Distance Calculation**

1. **Distance Formula:**
   * For stereo vision: Distance=f⋅Bd\text{Distance} = \frac{f \cdot B}{d}Distance=df⋅B​
     + fff is the focal length of the camera.
     + BBB is the baseline (distance between the two cameras).
     + ddd is the disparity (difference in pixel positions of the object between the two images).

#### Step 3: Calculating Steering Angle

##### **A. Object Position**

1. **Object Detection Output:**
   * Get the bounding box coordinates of detected objects (humans, tables, chairs).

##### **B. Calculate Centroid**

1. **Centroid Calculation:**
   * Compute the centroid (center point) of the bounding box: Centroid=(xmin+xmax2,ymin+ymax2)\text{Centroid} = \left( \frac{x\_{\text{min}} + x\_{\text{max}}}{2}, \frac{y\_{\text{min}} + y\_{\text{max}}}{2} \right)Centroid=(2xmin​+xmax​​,2ymin​+ymax​​)
     + xmin,yminx\_{\text{min}}, y\_{\text{min}}xmin​,ymin​ are the coordinates of the top-left corner.
     + xmax,ymaxx\_{\text{max}}, y\_{\text{max}}xmax​,ymax​ are the coordinates of the bottom-right corner.

##### **C. Steering Angle Calculation**

1. **Angle Calculation:**
   * Calculate the angle between the wheelchair's current heading and the target object's centroid.
   * Use trigonometry or a simple proportional controller to adjust the steering angle.
   * For example, if the centroid is to the right of the center of the image, turn right proportionally, and vice versa.

## Design Issues & limitation

Signal Interference: Wireless devices like the K10 Wireless Microphone are susceptible to interference from other electronic devices. This can lead to signal dropouts or degraded audio quality, disrupting the transmission of voice commands to the wheeling chair's control system and potentially causing delays or incorrect execution of commands.

Battery Life: The K10 Wireless Microphone offers up to 10 hours of continuous use. However, there is still a risk of the battery running out during critical times if not managed properly. Regular recharging and having spare microphones or backup solutions are necessary to ensure continuous operation.

Range Limitations: The effective range of the Bluetooth connection for the K10 Wireless Microphone can be limited, especially in larger environments or settings with multiple obstacles. Users must remain within the effective range to ensure reliable transmission of voice commands.

Environmental Noise: While the K10 Wireless Microphone features advanced noise cancellation, very noisy environments might still affect the clarity and accuracy of voice commands. This can lead to potential misinterpretations and incorrect actions by the control system of the wheeling chair.

Compatibility Issues: Ensuring that the K10 Wireless Microphone integrates seamlessly with the specific control system used in the wheeling chair is crucial. There may be specific requirements or configurations needed to ensure proper compatibility with the project's existing hardware and software.

Raspberry Pi Limitations: The Raspberry Pi used in this project has 4GB of RAM, which limits the ability to use more sophisticated models and techniques. The limited memory and processing power of the Raspberry Pi constrain the potential improvements and enhancements that could be made to the system.

Real-Time Processing Requirements:-

* Latency: Real-time processing is critical for ensuring safety and responsiveness. Delays in processing images or commands can lead to accidents or misinterpretations of the environment.
* Computational Power: High computational requirements for real-time image processing might demand powerful hardware, increasing the cost and power consumption.

#### Accuracy and Reliability of Obstacle Detection:-

* False Positives/Negatives: Inaccurate detection can lead to collisions or unnecessary stops. Ensuring high precision and recall in obstacle detection algorithms is crucial.
* Environmental Variability: Changes in lighting, weather conditions, or indoor/outdoor transitions can affect the reliability of the vision system.

#### Integration with Voice Commands:-

* Synchronization: Ensuring that voice commands and visual inputs are processed synchronously can be challenging. Delays in either system can result in unsafe operations.
* Context Awareness: The system needs to understand the context of voice commands relative to the visual environment (e.g., "go around that" requires understanding what "that" refers to).

#### Complexity of Visual Environments:-

* Dynamic Obstacles: Moving obstacles such as people or animals require advanced prediction and tracking algorithms.
* Static and Semi-Static Obstacles: Recognizing and navigating around fixed obstacles (furniture) and semi-static ones (doors that might open/close) demands robustness in detection and decision-making algorithms.

#### Hardware Limitations:-

* Camera Limitations: Quality and field of view of cameras can limit the effectiveness of the vision system. Low-resolution or narrow field of view cameras might miss critical details.
* Sensor Fusion: Integrating data from multiple sensors (e.g., LIDAR, ultrasonic sensors) along with vision can be complex but is often necessary for robust obstacle detection.

#### User Safety and Trust:-

* Fail-Safe Mechanisms: Ensuring that the system can safely handle failures in the vision system is crucial. This might include having manual override options or emergency stop mechanisms.
* User Trust: Users need to trust that the system will operate safely and reliably. This involves rigorous testing and validation.

#### Software and Algorithmic Challenges:-

* Algorithm Complexity: Advanced algorithms like YOLO, R-CNN, and SSD require significant computational resources. Optimizing these for real-time performance on embedded systems can be challenging.
* Edge Computing: Balancing the load between on-device processing and cloud-based processing, especially in areas with limited connectivity.

#### Testing and Validation:-

* Scalability of Testing: Extensive real-world testing in varied environments is needed to ensure reliability, which can be time-consuming and costly.
* Simulations: Developing realistic simulations for testing the vision and control systems before deployment.

#### Ethical and Privacy Concerns:-

* Data Privacy: Ensuring that any visual data captured is handled in compliance with privacy regulations.
* Ethical Use: Ensuring the system’s decisions are ethical, especially in environments with vulnerable individuals (elderly, disabled).

Addressing these design issues and limitations requires a multidisciplinary approach, combining advances in computer vision, real-time systems, user interface design, and robust testing methodologies.

*Chapter Five*

# Project Simulation and Performance Evaluation

#### 1. Simulation Environment

Creating a comprehensive simulation environment is essential for testing and evaluating the performance of the voice-controlled wheelchair. Here are key components:

* **Virtual Environment Setup**: Develop a virtual model of various real-world scenarios such as homes, hospitals, parks, and streets. Use simulation tools like Gazebo, Webots, or Unity.
* **Dynamic and Static Obstacle Placement**: Incorporate dynamic obstacles (moving objects like people and pets) and static obstacles (furniture, walls) in the simulation.
* **Voice Command Simulation**: Integrate a module to simulate voice commands and their interpretation. This could be done using speech recognition APIs.

#### 2. Simulation Scenarios

Design a variety of scenarios to test different aspects of the system:

* **Indoor Navigation**: Navigating through rooms, hallways, and around furniture.
* **Outdoor Navigation**: Navigating on sidewalks, through parks, and around obstacles like benches and trees.
* **Dynamic Obstacle Avoidance**: Handling scenarios with moving obstacles, such as crossing paths with pedestrians.
* **Voice Command Execution**: Testing how the system responds to a range of voice commands, including those that may require immediate action.

#### 3. Performance Metrics

Evaluate the system's performance based on several key metrics:

* **Response Time**: Measure the time taken for the system to process voice commands and visual data, and to execute actions.
* **Accuracy of Obstacle Detection**: Evaluate the precision and recall of the obstacle detection algorithms.
* **Command Execution Accuracy**: Assess how accurately the system executes voice commands, considering context and environment.
* **Battery Life**: Monitor the power consumption of the system, especially during continuous operation.
* **User Satisfaction**: Simulate user feedback and satisfaction levels based on the system's responsiveness and accuracy.

#### 4. Data Collection and Analysis

* **Log Data**: Collect detailed logs of the system’s operations, including command processing times, detected obstacles, and navigation paths.
* **Performance Analysis**: Use the collected data to analyze performance metrics and identify any patterns or areas needing improvement.
* **Error Analysis**: Investigate any errors or failures, understanding their causes and how they can be mitigated in future iterations.

#### 5. Iterative Testing and Refinement

* **Continuous Improvement**: Use insights from simulations to refine algorithms and system components iteratively.
* **Algorithm Optimization**: Focus on optimizing the computational efficiency of obstacle detection and navigation algorithms to enhance real-time performance.

#### 6. Field Testing

* **Controlled Environment Testing**: Before full deployment, test the system in controlled real-world environments to validate simulation results.
* **Pilot Studies**: Conduct pilot studies with actual users in real environments to gather feedback and further refine the system.
* **Safety Protocols**: Ensure robust safety protocols are in place during field testing to protect users.

#### 7. Evaluation Tools and Techniques

* **Benchmarking Tools**: Use benchmarking tools and software to compare the performance of different algorithms and system configurations.
* **Simulation Frameworks**: Utilize frameworks like ROS (Robot Operating System) for simulation and control of robotic systems.
* **Machine Learning Performance Evaluation**: Apply standard machine learning evaluation techniques to assess the performance of the vision-based algorithms.

#### 8. Documentation and Reporting

* **Comprehensive Reporting**: Document all simulation scenarios, testing procedures, performance metrics, and results.
* **User Manuals**: Develop detailed user manuals and guidelines for system operation and troubleshooting.

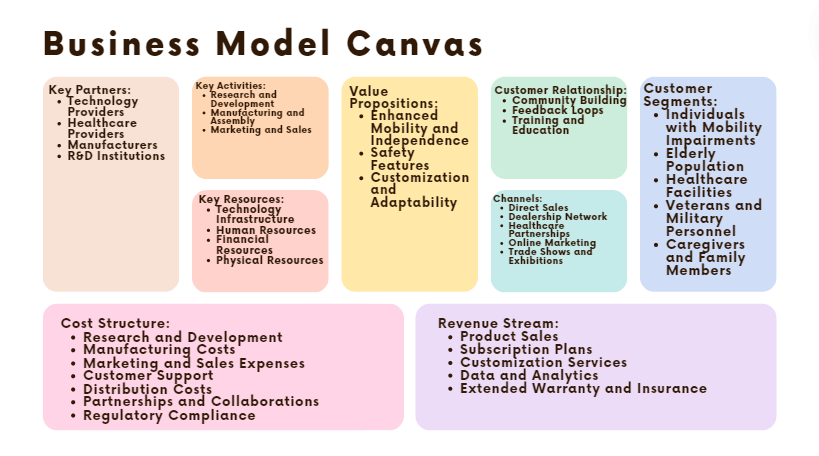
### Conclusion

The simulation and performance evaluation process for a voice-controlled wheelchair using computer vision is complex but essential. It ensures that the system is safe, reliable, and user-friendly before deployment in real-world environments. By meticulously planning and executing simulations, collecting and analyzing data, and iteratively refining the system, the final product can meet high standards of performance and safety.

*Chapter Six*

# Business Model

## Business Model Canvas



**Fig 6.1 business model**

## Components of the Business Model

**6.2.1 Key Partners**

**Universities and research organizations for continuous improvement and innovation. Companies specializing in voice recognition technology, AI, and smart sensors. Outsourcing production to established wheelchair manufacturers. Hospitals, clinics, and rehabilitation centers.**

**6.2.2 Key Activities**

**Continuous development of voice recognition technology and integration with wheelchair systems. Overseeing the production and assembly of the smart wheelchairs. Providing technical support, maintenance, and customization services.**

**6.2.3 Key Resources**

**Advanced voice recognition systems, AI algorithms, and smart sensors. Skilled R&D team, manufacturing experts, marketing and sales professionals, and customer support staff. Initial capital investment for R&D, production setup, and marketing. Manufacturing facilities, warehousing, and distribution network.**

**6.2.4 Value propositions**

**Allows users to navigate easily and independently using voice commands. Equipped with obstacle detection, automatic braking, and fall detection for user safety. Tailored solutions to meet individual needs and preferences.**

**6.2.5 Customer Relationship**

**Creating an online community for users to share experiences and tips. Regularly gathering user feedback for continuous improvement. Providing training sessions and resources to help users maximize the benefits of their smart wheelchair.**

**6.2.6 Channels**

**Company website with an online store for direct purchases. Authorized dealers and distributors for broader reach. Collaborations with hospitals, clinics, and rehabilitation centers to offer the product to patients. Digital marketing campaigns, social media, and content marketing to raise awareness. Participating in healthcare and technology trade shows to showcase the product.**

**6.2.7 Customer Segments**

**People with disabilities such as ALS, spinal cord injuries, and severe arthritis are the main target. Seniors who need mobility assistance to maintain their independence. Hospitals, rehabilitation centers, and nursing homes require advanced mobility solutions for their patients. Injured veterans who need specialized mobility aids. Those looking for reliable and advanced mobility solutions for their loved ones.**

**6.2.8 Cost Structure**

**Costs associated with developing and improving the voice recognition technology and smart wheelchair features. Production costs including raw materials, assembly, quality control, and labor. Costs for online and offline marketing campaigns, sales team salaries, and promotional activities. Expenses for maintaining a customer support team, providing training, and offering technical assistance. Warehousing, logistics, and shipping expenses. Costs related to establishing and maintaining relationships with key partners, including healthcare providers and insurance companies. Expenses for obtaining necessary certifications and ensuring compliance with healthcare regulations.**

|  |  |
| --- | --- |
| **Component** | **Price (EGP)** |
| **Arduino mega** | **1850** |
| **4 DC motor** | **300** |
| **H bridge** | **95** |
| **2 Ultrasonic** | **140** |
| **Buzzer** | **8** |
| **Aluminum Car body frame** | **800** |
| **Breadboard** | **100** |
| **AS5600** | **500** |
| **TCA9548** | **1000** |
| **Batteries + Bracket** | **400** |
| **Arduino Mega** | **1600** |
| **Raspberry PI 4 4GB RAM** | **6500** |
| **2 Cameras HD-3000** | **2400** |
| **Microphone K10** | **600** |
| **AS608** | **1500** |
| **DC DC converter** | **160** |
| **Wires** | **50** |
| **Power supply for Raspberry PI** | **1000** |
| **Total** | **19,003** |

**Table 6.1 cost structure**

**6.2.8 Revenue Stream**

**Revenue from direct sales of smart wheelchairs to consumers and healthcare facilities. Offering subscription-based services for regular maintenance, software updates, and tech support. Additional revenue from providing personalized customization options to users. Licensing anonymized usage data to healthcare providers and researchers for a fee. Offering extended warranty plans and insurance coverage for an additional fee.**

*Chapter Seven*

# Conclusion and Future Work

### 7.1 Conclusion

The development of a voice-controlled wheelchair represents a significant advancement in assistive technology, enhancing mobility and independence for individuals with disabilities. This project has successfully integrated advanced motor and camera systems with a robust voice recognition module, ensuring that users can control the wheelchair through intuitive voice commands. Utilizing a Raspberry Pi for audio processing and command recognition and an Arduino Mega for motor control, the system effectively translates spoken commands into precise movements, thereby improving user interaction and overall functionality.

Throughout the development process, several challenges were encountered and addressed. These included managing signal interference, ensuring sufficient battery life, dealing with range limitations, and mitigating environmental noise. Seamless integration of system components was essential to avoid operational glitches, and the limited processing power of the Raspberry Pi with 4GB of RAM necessitated careful optimization of the software, particularly the VOSK-Model-Small-en-us-0.15 for speech recognition.

### 7.2 Future Work

Future enhancements for the voice-controlled wheelchair project should focus on improving reliability, accuracy, and overall performance. Key areas for future development include:

1. **Voice Recognition System:**
   * Increase the accuracy of the voice recognition system to better handle variations in speech patterns, accents, and environmental noise.
   * Customize the VOSK-Model-Small-en-us-0.15 to recognize a wider range of commands and improve its ability to distinguish between similar-sounding words.
2. **Hardware Upgrades:**
   * Upgrade the hardware to support more resource-intensive applications. Future iterations could benefit from more powerful hardware to accommodate advanced features and enhance overall performance.
3. **Advanced Navigation Systems:**
   * Integrate more sophisticated navigation systems, such as advanced computer vision and machine learning algorithms, to improve the wheelchair's ability to navigate complex environments autonomously.
   * Implement real-time obstacle detection and path planning to ensure safe and efficient movement.
4. **User Feedback Mechanisms:**
   * Develop user feedback mechanisms to continuously monitor and adapt the system based on user interactions and preferences.
   * Create more intuitive user interfaces and ensure the system can learn from user behavior to provide a more personalized and responsive control experience.
5. **Alternative Models for Speech Recognition:**
   * Explore other models that might offer better performance, such as the DeepSpeech model developed by Mozilla. DeepSpeech is known for its high accuracy and ability to function offline, making it suitable for real-time applications. It has robust noise filtering capabilities and is designed to handle diverse accents and speech variations effectively.

**For mechanical design future work**:

Implementing a wheelchair in a mechanical way using a BLDC (Brushless DC) motor controlled by an ESC (Electronic Speed Controller) involves a multifaceted approach, incorporating various engineering principles and tools. The key components of this project include the selection and integration of the BLDC motor, the development of a PID (Proportional-Integral-Derivative) control system for precise motor control, and conducting stress analysis and design simulations using Simulink.

### BLDC Motor and ESC Integration

The BLDC motor is chosen for its high efficiency, reliability, and performance. It operates through electronic commutation, which is managed by the ESC. The ESC not only controls the speed and direction of the motor but also ensures smooth and precise operation, which is crucial for a wheelchair's functionality. Proper integration involves selecting an appropriate motor and ESC based on the required torque, speed, and load capacity of the wheelchair.

### PID Control System

The PID control system is essential for maintaining the desired speed and direction of the wheelchair. It adjusts the motor's input based on the feedback from sensors, ensuring stability and responsiveness. The PID controller continually calculates an error value as the difference between a desired setpoint and a measured process variable, applying corrections based on proportional, integral, and derivative terms. This control mechanism helps in achieving smooth acceleration, deceleration, and precise maneuvering of the wheelchair.

### Simulink Implementation

Simulink provides a powerful platform for modeling, simulating, and analyzing the control systems and mechanical structures involved in the wheelchair design. By creating a virtual prototype in Simulink, engineers can test and refine the PID control system, simulate the interaction between the motor and the mechanical components, and conduct stress analysis under various operating conditions. This approach reduces the need for physical prototypes, saving time and resources while improving the overall design process

### Stress Analysis and Design

Stress analysis is a critical part of the wheelchair design process to ensure the structure can withstand the various loads and forces it will encounter during operation. Using Simulink, a model-based design approach allows for simulating and analyzing the mechanical stresses on the wheelchair frame and components. This helps in identifying potential points of failure and optimizing the design for durability and safety. Simulink also aids in visualizing the dynamic performance of the system, making it easier to iterate and improve the design

# bldc

The BLDC motor is widely used in applications including appliances, automotive, aerospace, consumer, medical, automated industrial equipment and instrumentation.

The BLDC motor is electrically commutated by power switches instead of brushes. Compared with a brushed DC motor or an induction motor, the BLDC motor has many advantages:[1]

* Higher efficiency and reliability
* Lower acoustic noise
* Smaller and lighter
* Greater dynamic response
* Better speed versus torque characteristics
* Higher speed range
* Longer life

This document initially provides a general overview to familiarize the reader with motor control fundamentals, terms and concepts, and applications. The latter portion of this document provides detailed descriptions of motor structure, working principle, characteristics and control methods.

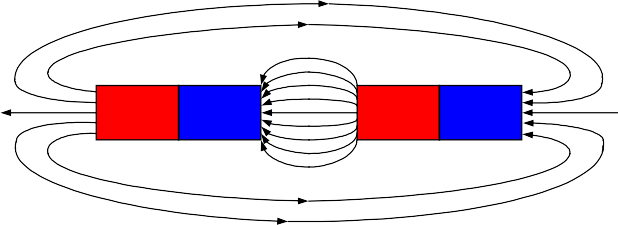
# MOTOR FUNDAMENTAL CONCEPTS

### General Motor Principles

Motors convert electrical energy into mechanical energy using electromagnetic principles. The energy conversion method is fundamentally the same in all electric motors. This document starts with a general overview of basic electromagnetic physics before entering discussing the details of motor operation.

### Magnetic Force

Magnetic poles generate invisible lines of magnetic force flowing from the north pole to the south pole as shown in [Figure 1](#_bookmark5). When magnetic poles of opposite polarity face each other, they generate an attractive force, while like poles generate a repulsive force.

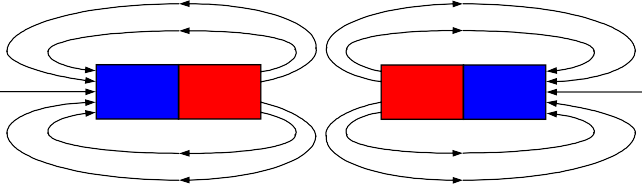


**N**

**S**

**N**

**S**



**S**

**N**

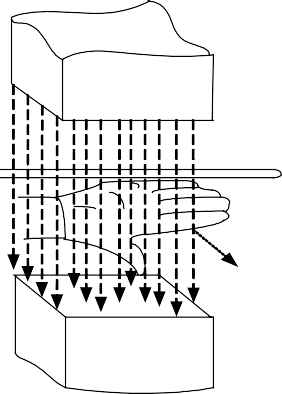
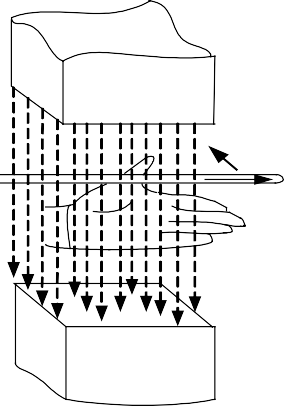
**N**

**S**

### Unlike-pole attraction (b) Like-pole Fig 7.1 repulsion —Magnetic Force

### Left-Hand Rule

Current in a conductor generates a magnetic field. Placing a conductor in the vicinity of a separate magnetic can generate a force that reaches its apex when the conductor is at 90° to the external field. The left-hand rule can help the user determine the direction of the force, as shown in [Figure 2](#_bookmark7)(a).

Left-Hand Rule: Extend the left hand with the thumb and four fingers on the same plane with the thumb pointing out. Face the palm towards the north pole of the external magnetic field and the four fingers in the direction of the current; the thumb points in the direction of the force.

|  |  |
| --- | --- |
| N  ***F***  ***i***  ***B***  S | N  ***- +***  ***e***  ***v***  ***B***  S |
| **(a) Left-Hand Rule** | **(b) Right-Hand Rule** |

### Fig 7.2—Left-Hand Rule and Right-Hand Rule

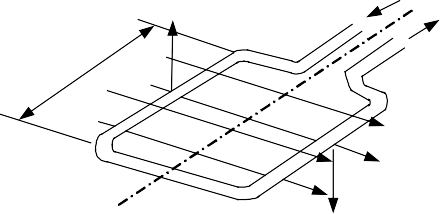
The magnitude of the force can be calculated from the equation below:

## F  BIL sin  (1)

Where F is the electromagnetic force, B is the magnetic field density, I is the conductor current, L is the length of the conductor, and θ is the angular difference between B and I.

Given that a coil usually has two effective conductors: *a-b* and *c-d* shown in [Figure 3](#_bookmark8)(a), these two conductors induce two forces of opposite direction when current passes through in the magnetic field.

O’



*F*

***i***

a

O’

*L*

d

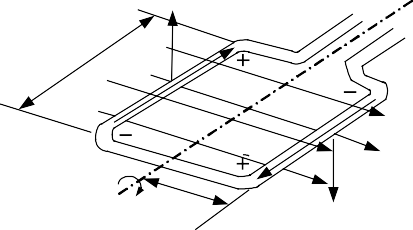
b

*B*

O

c

*F*



*v*

a

*L*

e

d

e

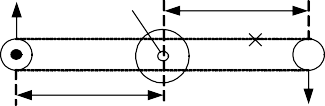
b

*B*

O *? r* c

*v*

*F*



*OO*’

r

r *F*

### (a) (b) (c) —Coil in a Magnetic Field

The torque is the product of the tangential force acting at a radius with units of force multiplied by length. If there are N continuous coil turns, and based on the parameters in [Figure 3](#_bookmark8)(b), the generated torque equals:

Where:

## TD  2rFN  2rBILN  KTI

(2)

### Right-Hand Rule

The movement of the conductor in the magnetic field induces an electromotive force known as the BEMF. The right-hand rule can determine the direction of the force as shown in [Figure 2](#_bookmark7)(b).

The Right-Hand Rule: Stretch out the right hand with the four fingers and the thumb on the same plane,the palm facing the north pole of the external magnetic field, and the thumb pointing in the direction of the velocity of v. The four fingers point in the direction of the induced electromotive force.

The magnitude of the induced electromotive force can be calculated as:

## E  BLv sin  (3)

Where:

E is the induced electromagnetic force (V). v is the velocity of the conductor (m/s).

θ is the angular difference between B and L (rad).

When the motor rotates at an angular velocity of ω (rad/s) and there are N coil turns, the total electromotive force is:

## E  2BLvN  2BLrN  KE (4)

Where:

ω is the angular velocity (rad/s).

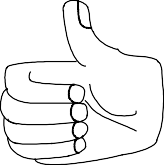
r is the internal radius of the motor (m).

KE=2rBLN is the electromotive force constant (V·s/rad). Based on the parameters from [Figure 3](#_bookmark8)(c)

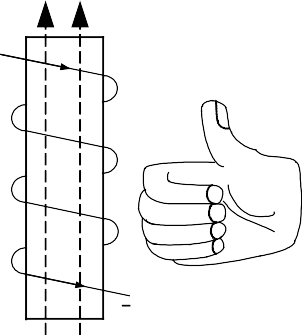
### Right-Hand Corkscrew Rule

Given that an electrical current flowing in a straight line generates a magnetic field as shown in Figure 4(a) coiling the conductor would therefore generate clear magnetic poles as shown in Figure 4(b), with the direction of the magnetic fields determined by the right-hand corkscrew rule.

Right-Hand Corkscrew Rule: For a current flowing in a straight line as shown in Figure 4(a), the thumb points in the direction of the current I, and the fingers curl in the direction of the magnetic field B. For a coiled current as shown in Figure 4(b), the fingers curl in the direction of the current I, and then the thumb points in the direction of the magnetic field B through the center of the loop.



**i**



**N**

***i***

**S**

**(a) Straight line (b) Loop**

**Fig 7.3—Right-Hand Corkscrew Rule**

### Stator

There are three classifications of the BLDC motor: single-phase, two-phase and three-phase. This discussion assumes that the stator for each type has the same number of windings. The single-phase and three-phase motors are the most widely used. [Figure 5](#_bookmark12) shows the simplified cross section of a single-phase and a three-phase BLDC motor. The rotor has permanent magnets to form 2 magnetic pole pairs, and surrounds the stator, which has the windings.

|  |  |
| --- | --- |
| **Stator**  **N Rotor**  **A**  **Air gap**  **B**  **S S**  **Permanent**  **N magnets** | **Stator**  **N**  **Rotor**  **A**  **B C**  **S S Air gap**  **Permane**  **N nt magnet** |
| **(a) Single-phase** | **(b) Three-phase** |

### Fig 7.4—Simplified BLDC Motor Diagrams

A single-phase motor has one stator winding—wound either clockwise or counter-clockwise along each arm of the stator—to produce four magnetic poles as shown in [Figure 5](#_bookmark12)(a). By comparison, a three- phase motor has three windings as shown in [Figure 5](#_bookmark12)(b). Each phase turns on sequentially to make the rotor revolve.

There are two types of stator windings: trapezoidal and sinusoidal, which refers to the shape of the back electromotive force (BEMF) signal. The shape of the BEMF is determined by different coil interconnections and the distance of the air gap. In addition to the BEMF, the phase current also follows a trapezoidal and sinusoidal shape. A sinusoidal motor produces smoother electromagnetic torque than a trapezoidal motor, though at a higher cost due to their use of extra copper windings. A BLDC motor uses a simplified structure with trapezoidal stator windings.

### Rotor

A rotor consists of a shaft and a hub with permanent magnets arranged to form between two to eight pole pairs that alternate between north and south poles. [Figure 6](#_bookmark14) shows cross sections of three kinds of magnets arrangements in a rotor.

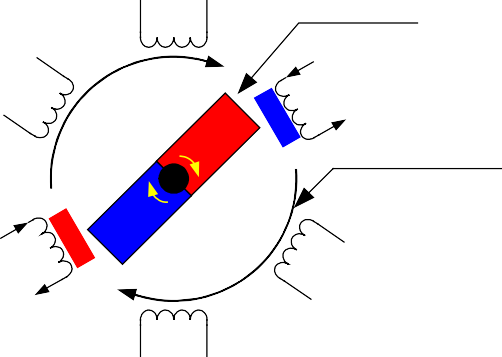
There are multiple magnet materials, such as ferrous mixtures and rare-earth alloys. Ferrite magnets are traditional and relatively inexpensive, though rare-earth alloy magnets are becoming increasingly popular because of their high magnetic density. The higher density helps to shrink rotors while maintaining high relative torque when compared to similar ferrite magnets.

|  |  |  |
| --- | --- | --- |
| **Hub**  **Shaft N N**  **S N**  **S S**  **S S**  **N N**  **Permanent**  **magnets N N**  **S S**  **N S**  **S N** | | |
| **(a) Surface-Mounted** | **(b) Embedded** | **(c) Inserted** |

### Fig 7.5—Rotor Magnets Cross-Sections

* 1. **Operational Motor Theory**

Motor operation is based on the attraction or repulsion between magnetic poles. Using the three-phase motor shown in [Figure 7](#_bookmark17), the process starts when current flows through one of the three stator windings and generates a magnetic pole that attracts the closest permanent magnet of the opposite pole. The rotor will move if the current shifts to an adjacent winding. Sequentially charging each winding will cause the rotor to follow in a rotating field. The torque in this example depends on the current amplitude and the number of turns on the stator windings, the strength and the size of the permanent magnets, the air gap between the rotor and the windings, and the length of the rotating arm.



**A1**

**Magnet bar**

**B2**

**C2**

**S**

**N**

**Rotating magnetic field**

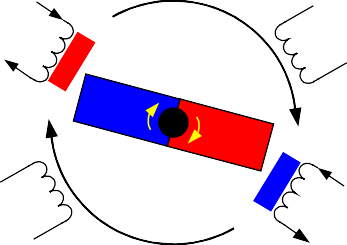
**S**

**N**

**C1**

**B1**

**A2**



**A1**

**B2**

**C2**

**N**

**S**

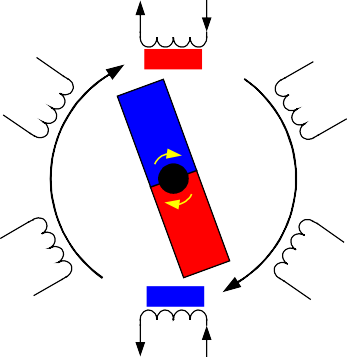
**N**

**S**

**C1**

**B1**

**A2**



**A1**

**N**

**B2**

**C2**

**S**

**N**

**C1**

**B1**

**S**

**A2**

# MOTOR VARIETIES

### Fifigure 7.6—Motor Rotation

There are multiple varieties of electric motor differentiated by structure and signal type, but are generally based on the same principle as the three-phase motor previously discussed. [Figure 8[](#_bookmark18)2] diagrams the different motors organized by classifying features.

Series Compound

Wound Field

PM

Homopolar

Commutator

DC

**Electric Motors**

AC

Asynchronous

Synchronous

Induction

Shunt

Single Phase

Poly- phase

Reluctance

Hysteresis

Stepper

Brushless DC

Sinusoidal

### Figure7.7—Motor Classification

The primary difference between AC and DC motors is the power type applied to the armature. From this vantage, a BLDC motor actually is an AC motor. The difference between an asynchronous and a synchronous motor is whether or not the rotor runs at the same frequency as the stator. Each motor favors specific applications. [Figure 9](#_bookmark25) illustrates some of the more popular motor designs.

### Introduction to Various Motor Types

### Brushed DC Motor

A brushed DC motor consists of a commutator and brushes that convert a DC current in an armature coil to an AC current, as shown in [Figure 9](#_bookmark25)(a). As current flows through the commutator through the armature windings, the electromagnetic field repels the nearby magnets with the same polarity, and causes the winging to turn to the attracting magnets of opposite polarity. As the armature turns, the commutator reverses the current in the armature coil to repel the nearby magnets, thus causing the motor to continuously turn. The fact that this motor can be driven by DC voltages and currents makes it very attractive for low cost applications. However, the arcing produced by the armature coils on the brush-commutator surface generates heat, wear, and EMI, and is a major drawback.

### Brushless DC (BLDC) Motor

A BLDC motor accomplishes commutation electronically using rotor position feedback to determine when to switch the current. The structure is shown in [Figure 9](#_bookmark25)(b). Feedback usually entails an attached Hall sensor or a rotary encoder.

The stator windings work in conjunction with permanent magnets on the rotor to generate a nearly uniform flux density in the air gap. This permits the stator coils to be driven by a constant DC voltage (hence the name brushless DC), which simply switches from one stator coil to the next to generate an AC voltage waveform with a trapezoidal shape.

### AC Induction Motor (ACIM)

A sinusoidal AC current runs through the stator to create a rotating variable magnetic field that induces a current in the rotor (typically made of non-ferrous materials). This induced current circulates in the bars of the rotor to generate a magnetic field. These two magnetic fields run at different frequencies (usually ω-s>ω-r for the motor) and to generate torque. [Figure 9](#_bookmark25)(c) shows the motor structure.

### Permanent Magnet Synchronous Motor (PMSM)

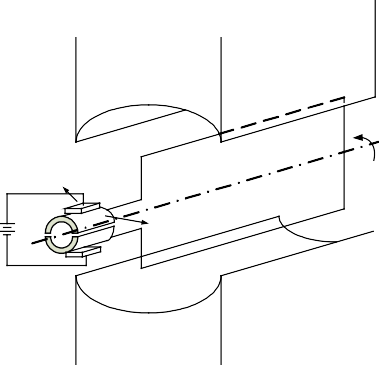
The PMSM motor shares some similarities with the BLDC motor, but is driven by a sinusoidal signal to achieve lower torque ripple. The sinusoidal distribution of the multi-phase stator windings generates a sinusoidal flux density in the air gap that is different from BLDC motor’s trapezoidal flux density. However, newer designs can achieve this sinusoidal flux density with concentrated stator windings and a modified rotor structure. Rotor magnet position can significantly alter the electrical properties of a PMSM; Mounting the rotor magnets on the surface—as shown in [Figure 6](#_bookmark14)(a)—results in lower torque ripple, while burying the magnets inside the rotor structure—as shown in [Figure 6](#_bookmark14)(b)—increases saliency, which increases the reluctance torque of the motor. The structure of PMSM is shown in [Figure](#_bookmark25) [9](#_bookmark25)(d).

### Stepper Motor & Switched Reluctance (SR) Motor

Both stepper motors and SR motors have similar physical structures; The stator consists of concentrated winding coils while the rotor is made of soft iron laminates without coils. It has a doubly salient structure (teeth on both the rotor and stator) as shown in [Figure 9](#_bookmark25)(e).

Stepper motors are designed to replace more expensive servo motors. When the current switches from one set of stator coils to the next, the magnetic attraction between rotor and stator teeth induces enough torque to rotate the rotor to the next stable position, or "step." The rotation speed is determined by the frequency of the current pulse, and the rotational distance is determined by the number of pulses. Since each step results in a small displacement, a stepper motor is typically limited to low-speed position-control applications.

There is no reactive torque (magnet to magnet) in an SR motor because the rotor cannot generate its own magnetic field. Instead, both rotor and stator poles have protrusions so that the flux length is a function of angular position, which gives rise to saliency torque. This is the only torque-producing mechanism in an SR motor, which tends to result in high torque ripple. However, due to their simple design, SR motor is very economical to build, and is perhaps the most robust motor available.



**N**

**Brushes**

**Commutator**

**S**



**N**

**SW1**

**SW3**

**OUT1 OUT2**

**S**

**S**

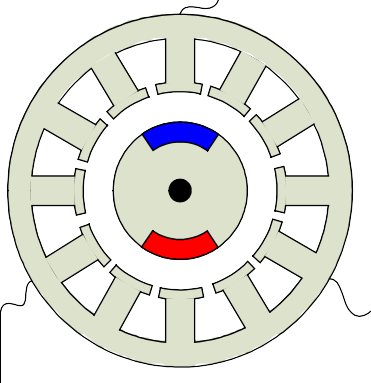
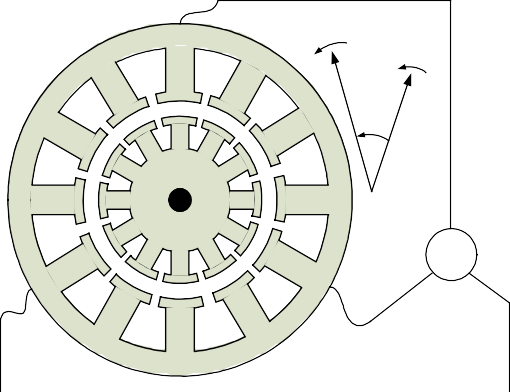
**SW2**

**SW4**

**N**

(a) Brushed DC motor (b) Brushless DC (BLDC) motor

**U ? -s U**



**? -r**

**Slip**

**S**

**Three phase**

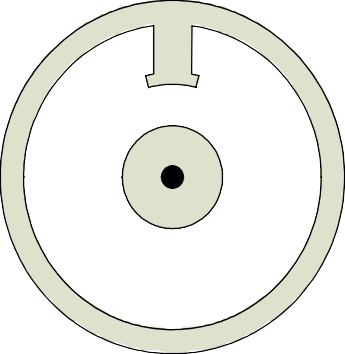
**AC AC power N**

**V W V W**

**Three phase AC AC power**

(c) AC induction motor (ACIM) (d) Permanent magnet synchronous motor (PMSM)

**Stepper motor control**



**Laminated rotor**

**Phase winding**

**SR motor control**

(e) Stepper motor & Switched reluctance (SR) motor

### Figure7.8—Structures of Different Types of Motors

* 1. **Comparison of Various Motor Types**

The BLDC motor has several advantages over other motors. [Table 1](#_bookmark27) and [Table 2](#_bookmark28) summarize the advantages of the BLDC motor when compared against a brushed DC motor and an AC induction motor.[1][3]

### Table 7 — Comparison between BLDC motor and brushed DC motor

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **BLDC Motor** | **Brushed DC Motor** | **Actual Advantage** |
| Commutation | Electronic commutation based on rotor position information | Mechanical brushes and commutator | Electronic switches replace the mechanical devices |
| Efficiency | High | Moderate | Voltage drop on electronic device is smaller than that on brushes |
| Maintenance | Little/None | Periodic | No brushes/commutator maintenance. |
| Thermal performance | Better | Poor | Only the armature windings generate heat, which is the stator and is connected to the outside case of the BLDC.;The case dissipates heat better than a rotor located inside of brushed DC motor. |
| Output Power/ Frame Size (Ratio) | High | Moderate/Low | Modern permanent magnet and no rotor losses. |
| Speed/Torque Characteristics | Flat | Moderately flat | No brush friction to reduce useful torque. |
| Dynamic Response | Fast | Slow | Lower rotor inertia because of permanent magnets. |
| Speed Range | High | Low | No mechanical limitation imposed by brushes or commutator |
| Electric Noise | Low | High | No arcs from brushes to generate noise, causing EMI problems. |
| Lifetime | Long | Short | No brushes and commutator |

**Table 8—Comparison between BLDC Motor and AC Induction Motor**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **BLDC motor** | **AC induction motor** | **Actual Advantage** |
| Speed/Torque Characteristics | Flat | Nonlinear — lower torque at lower speeds | Permanent magnet design with rotor position feedback gives BLDC higher starting and low-speed torque |
| Output Power/ Frame Size (Ratio) | High | Moderate | Both stator and rotor have windings for induction motor |
| Dynamic Response | Fast | Low | Lower rotor inertia because of permanent magnet |
| Slip Between Stator And Rotor Frequency | No | Yes; rotor runs at a lower frequency than stator by slip frequency and slip increases with load on the motor | BLDC is a synchronous motor, induction motor is an asynchronous motor |

The primary disadvantage of BLDC is cost, though this is no inherent reason due to the motor itself; the construction of a BLDC motor is actually simpler than that of brushed DC motor or AC induction motor. The higher cost of BLDC motor is caused by the additional driver circuit for BLDC motor. However if the application requires adjustable speed, accurate position control, or requires a driver circuit, then BLDC motor is not only advantageous but also less expensive overall.

# BRUSHLESS DC MOTOR CONTROL

### Switch Configuration and PWM

Brushless DC motors use electric switches to realize current commutation, and thus continuously rotate the motor. These electric switches are usually connected in an H-bridge structure for a single-phase BLDC motor, and a three-phase bridge structure for a three-phase BLDC motor shown in [Figure 10](#_bookmark31). Usually the high-side switches are controlled using pulse-width modulation (PWM), which converts a DC voltage into a modulated voltage, which easily and efficiently limits the startup current, control speed and torque. Generally, raising the switching frequency increases PWM losses, though lowering the switching frequency limits the system’s bandwidth and can raise the ripple current pulses to the points where they become destructive or shut down the BLDC motor driver.[4]

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Three-phase Brushless DC Motor  SW1 Single-phase SW3 SW1 SW3 SW5 **U**  Brushless DC Motor  **U**  **M M**  **OUT1 OUT2 V W V W**  SW2 SW4 SW2 SW4 SW6 | | | | | | | | |
| **Figure 7.9 H-bridge** |  |  |  |  |  |  |  | **Figure 7.10 Three-phase bridge** |

### Electronics Commutation Principle

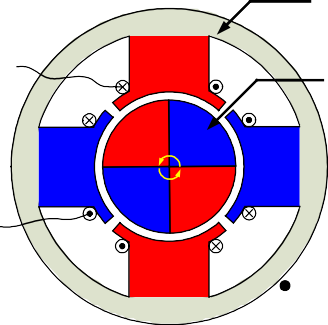
### Single-Phase BLDC Motor

BLDC commutation relies on feedback on the rotor position to decide when to energize the corresponding switches to generate the biggest torque. The easiest way to accurately detect position is to use a position sensor. The most popular position sensor device is Hall sensor. Most BLDC motors have Hall sensors embedded into the stator on the non-driving end of the motor.

[Figure 11](#_bookmark34) shows the commutation sequence of a single-phase BLDC motor driver circuit. The permanent magnets form the rotor and are located inside the stator. A Hall position sensor (“a”) is mounted to the outside stator, which induces an output voltage proportional to the magnetic intensity (assume the sensor goes HIGH when the rotor’s North Pole passes by, and goes LOW when the rotor’s South Pole passes by). SW1 and SW4 turn on when Hall sensor output is HIGH, as shown in [Figure 11](#_bookmark34)(a) and (b). At this stage, armature current flows through the stator windings from OUT1 to OUT2 and induces the alternate stator electromagnetic poles accordingly. The magnetic force generated by rotor magnetic field and stator electromagnetic field causes the rotor to rotate. After the rotor signal reaches 180°, the Hall output voltage reverses due to its proximity to a South Pole. SW2 and SW3 then turn on with current reversing from OUT2 to OUT1, as shown in [Figure 11](#_bookmark34)(c) and (d). The opposite stator magnetic poles induce the rotor to continue rotating in the same direction.

Figure 7.11 shows an example of Hall sensor signals with respect to switch drive signals and armature current.[5] The armature current exhibits a saw tooth waveform due to PWM control. The applied voltage, switching frequency, and the PWM duty cycle are three key parameters to determine the speed and the torque of the motor.

**Stator**



**OUT1**

**N**

**Rotor**

**N S**

**S**

**S**

**S N**

**OUT2**

**N**

**a**



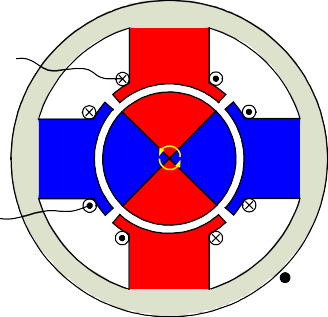
**SW1 SW3**

**i**

**OUT1 M OUT2**

**SW2**

**SW4**



**OUT1**

**N**

**N**

**S**

**S**

**S**

**S**

**N**

**OUT2**

**N**

**a**



**SW1 SW3**

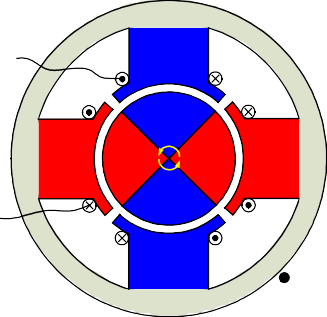
**i**

**OUT1 M OUT2**

**SW2**

**SW4**

* 1. Hall sensor value: a=1 (a) Hall sensor value: a=1 (from 0)



**OUT1**

**S**

**S**

**N**

**N**

**N**

**N**

**S**

**OUT2**

**S**

**a**



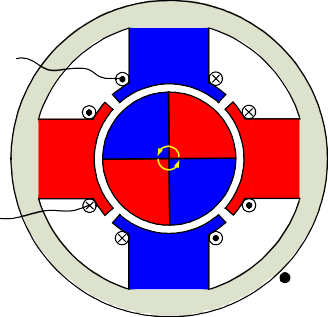
**SW1 SW3**

**i**

**OUT1 M OUT2**

**SW2**

**SW4**



**OUT1**

**S**

**S N**

**N**

**N**

**N S**

**OUT2**

**S**

**a**



**SW1 SW3**

**i**

**OUT1 M OUT2**

**SW2**

**SW4**

* 1. Hall sensor value: a=0 (from 1) (d) Hall sensor value:

**Figure 7.11—Single-Phase BLDC Motor Commutation Sequence**

Hall



1

0

0

SW1

SW2

SW3

SW4

IOUT1-2

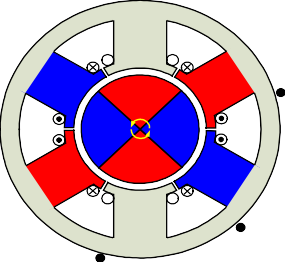
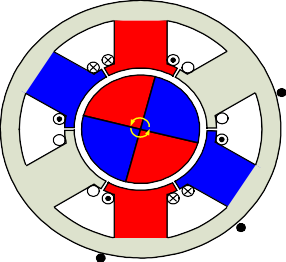
### Figure 7.12—Single-Phase BLDC Motor Sensor versus Drive Timing

### Three-Phase BLDC Motor

A three-phase BLDC motor requires three Hall sensors to detect the rotor’s position. Based on the physical position of the Hall sensors, there are two types of output: a 60° phase shift and a 120° phase shift. Combining these three Hall sensor signals can determine the exact communation sequence.

Figure 7.12 shows the commutation sequence of a three-phase BLDC motor driver circuit for counter- clockwise rotation. Three Hall sensors—“a,” “b,” and “c”—are mounted on the stator at 120° intervals, while the three phase windings are in a star formation. For every 60° rotation, one of the Hall sensors changes its state; it takes six steps to complete a whole electrical cycle. In synchronous mode, the phase current switching updates every 60°. For each step, there is one motor terminal driven high, another motor terminal driven low, with the third one left floating. Individual drive controls for the high and low drivers permit high drive, low drive, and floating drive at each motor terminal.

However, one signal cycle may not correspond to a complete mechanical revolution. The number of signal cycles to complete a mechanical rotation is determined by the number of rotor pole pairs. Every rotor pole pair requires one signal cycle in one mechanical rotation. So, the number of signal cycles is equal to the rotor pole pairs.

**U U**

**N**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**V W V W**

**S S N**

**N a N a**

**S**

**S S**

**S**

**N N**

**S N S**

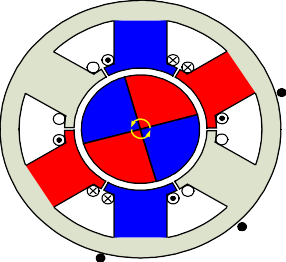
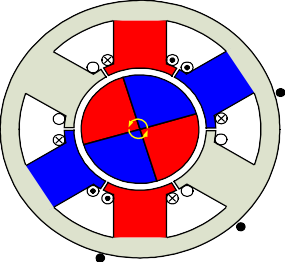
**W V W V N**

**U c U c**

**b b**

Hall sensor value: abc=001 Hall sensor value: abc=011

**U U**



**N S**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**V W V W**

**S N**

**S a N a**

**N S**

**N S**

**S N**

**S N**

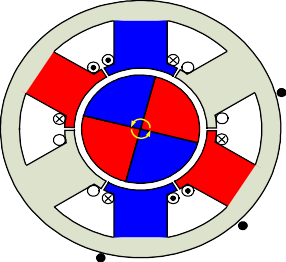
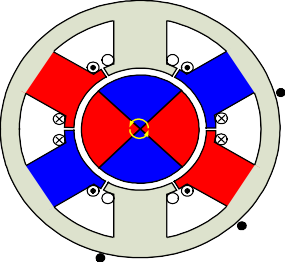
**W V W V N S**

**U c U c**

**b b**

Hall sensor value: abc=101 Hall sensor value: abc=010

**U U**



**S**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**SW1 SW3 SW5**

**U**

**M**

**V W**

**SW2 SW4 SW6**

**V W V W**

**N S N**

**S a S a**

**N**

**N N**

**N**

**S S**

**S N N**

**W V W V**

**S**

### Figure 7.13 —Three-Phase BLDC Motor Commutation Sequence

[Figure 7.13](#_bookmark38) shows the timing diagrams where the phase windings—U, V, and W—are either energized or floated based on the Hall sensor signals a, b, and c. This is an example of Hall sensor signal having a 120° phase shift with respect to each other, where the motor rotates counter-clockwise. Producing a Hall signal with a 60° phase shift or rotating the motor clockwise requires a different timing sequence. To vary the rotation speed, use pulse width modulation signals on the switches at a much higher frequency than the motor rotation frequency. Generally, the PWM frequency should be at least 10 times higher than the maximum motor rotation frequency. Another advantage of PWM is that if the DC bus voltage is much higher than the motor-rated voltage, so limiting the duty cycle of PWM to meet the motor rated voltage controls the motor.

a



b c

U

V

W

Hall Sensor Code

001

101 100 110 010 011 001 101 100 110 010 011

Electrical cycle Electrical cycle One mechanical rotation

001

High Float Low

High Float Low High Float Low

### Figure 7.14—Three-phase BLDC motor sensor versus drive timing

1. **Sensorless BLDC Motor Control**

However, sensors cannot be used in applications where the rotor is in a closed housing and requires minimal electrical entries, such as a compressor or applications where the motor is immersed in a liquid. Therefore, the BLDC sensorless driver monitors the BEMF signals instead of the position detected by Hall sensors to commutate the signal. The relationship between the sensors’ output and the BEMF is shown in [Figure 15](#_bookmark40). The sensor signal changes state when the voltage polarity of the BEMF crosses from positive to negative or from negative to positive. The BEMF zero-crossings provides precise position data for commutation.[6]

However, as BEMF is proportional to the speed of rotation, this implies that the motor requires a minimum speed for precise feedback. So under very low speed conditions—such as start-up— additional detectors—such as open loop or BEMF amplifiers—are required to control the motor (This is beyond the scope of this application note).

The sensorless commutation can simplify the motor structure and lower the motor cost. Applications in dusty or oily environments that require only occasional cleaning, or where the motor is generally inaccessible, benefit from sensorless communation.

a



b c

BEMF

U-V

V-W

W-U

Hall Sensor Code

001 101 100 110 010 011 001 101 100 110 010 011

Electrical cycle Electrical cycle One mechanical rotation

### Figure 7.15—Hall Sensor versus BEMF

+ 0

-

+

0

-

+ 0

-

001

# SUMMARY

This application note introduces the motor fundamentals, with special attention to BLDC motors.. As described in this document, a BLDC motor has many advantages over a brushed DC motor and an AC induction motor: It is easily controlled with position feedback sensors and generally performs well, especially in speed/torque. With these advantages, BLDC motor will spread to more applications. Moreover, with the development of sensorless technology, BLDC motor will become convenient or indispensable in applications with environmental limitations.

## DRIVING MOTOR

Brushless motors typically have a higher efficiency than their brushed counterparts, which means they can offer more power and speed with less energy consumption. They also tend to be quieter and require less maintenance since they don't have brushes that wear down over time.

****

**Figure 7.16**

#### SPECIFICATIONS

##### For 36 volt &4.4A/h

* Power output: 250-350 watts
* Voltage: 36-42 volts
* Max speed: 10-15 mph (16-24 km/h)
* Max torque: 15-20 Nm
* Efficiency: 80-90%
* Motor weight: 1.5-2.5 kg

#### SPECIFICATIONS

##### For 12 volt &7A/h

* Power output: 150-200 watts
* Max speed: 5-10 mph (8-16 km/h)
* Max torque: 7-10 Nm
* Efficiency: 70-80%

The motor driver used is (ESC 30A BLDC) and it’s not suitable for the car because it takes only 12 volts which leads to experiencing poor starting torque with 8.5-inch brushless motor and rotate the motor with only one direction so we fixed that by using relays to switch one phase from 3 phases to drive the motor with both direction.

figure 7.17

Increasing the voltage supplied to the motor can also increase the torque output, up to a certain point. This is because higher voltages can cause the motor to spin faster, which can increase the back-EMF (electromotive force) generated by the motor.As the suitable motor driver for this motor is [O-Drive v3.6]

The O-Drive v3.6 is a high-performance motor controller designed for use with brushless motors. Some of the key specifications of the O-Drive v3.6 include:

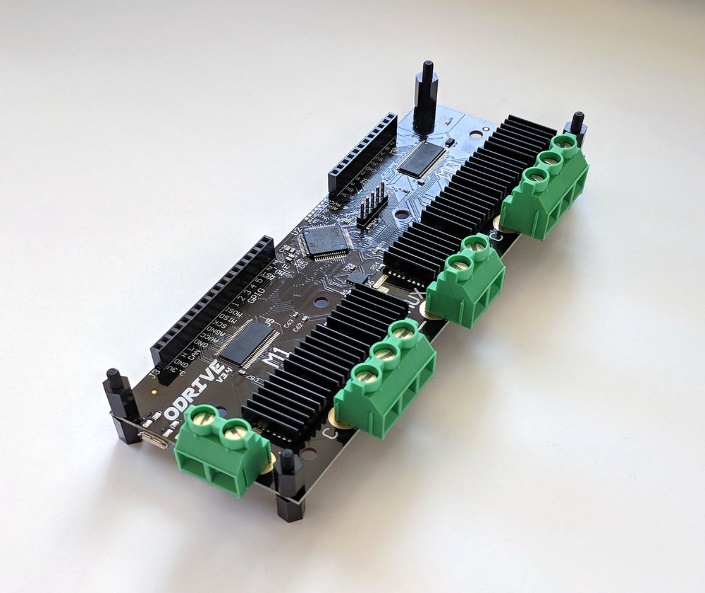
1. Motor compatibility: The O-Drive v3.6 is compatible with a wide range of brushless motor types, including AC induction motors, permanent magnet synchronous motors, and brushless DC motors.
2. Input voltage: The O-Drive v3.6 can accept input voltages ranging from 12 to 56 volts DC.
3. Peak current: The O-Drive v3.6 can handle peak current levels of up to 75 amps, making it suitable for high-performance applications.
4. Continuous current: The O-Drive v3.6 is designed to handle continuous current levels of up to 30 amps, although this can be increased to up to 75 amps with additional cooling.
5. Encoder support: The O-Drive v3.6 supports a wide range of encoder types, including incremental encoders, absolute encoders, and hall-effect sensors.

Figure 7.18

We could not use this driver because of his high price and it’s unavailability in the country, therefore if we requested it would arrive late. Although it can control the motor in both direction

# 30A BLDC ESC



Figure 7.19

This is fully programmable 30A BLDC ESC with 5V, 3A BEC. Can drive motors with continuous 30Amp load current. It has sturdy construction with 2 separate PCBs for Controller and ESC power MOSFETs. It can be powered with 2-4 lithium Polymer batteries or 5-12 NiMH / NiCd batteries. It has separate voltage regulator for the microcontroller for providing good anti-jamming capability. It is most suitable for UAVs, Aircrafts and Helicopters.

## Specifications

* Output: 30A continuous; 40Amps for 10 seconds
* Input voltage: 2-4 cells Lithium Polymer / Lithium Ion battery or 5-12 cells NiMH / NiCd
* BEC: 5V, 3Amp for external receiver and servos
* Max Speed: 2 Pole: 210,000rpm; 6 Pole: 70,000rpm; 12 Pole: 35,000rpm
* Weight: 32gms
* Size: 55mm x 26mm x 13mm

## Features:

* High-quality MOSFETs for BLDC motor drive
* High-performance microcontroller for best compatibility with all types of motors at greater efficiency
* Fully programmable with any standard RC remote control
* Heat sink with high-performance heat transmission membrane for better thermal management
* Smooth, Linear, and Precise throttle response
* Low-Voltage cut-off protection
* Over-heat protection
* Separate voltage regulator IC for the microcontroller to anti-jamming capability
* Supported Motor Speed (Maximum): 210000RPM (2 poles), 70000RPM (6poles), 35000RPM (12 poles)

## Connections:

BLDC ESC has three Blue wires coming out from the one end which are to be connected to the BLDC motor. On the other end, it has red and black wires which are to be connected to the battery. It also has a 3-pin servo connector which is used for receiving the throttle command and for giving out regulated 5V, 3Amp supply for the remote receiver and the servo motors.

Table 9—ESC Connections

|  |  |  |
| --- | --- | --- |
| **Connection type** | **Wire Colour** | **Function** |
| Power | Red | 7.4 to 14.8V |
|  | Black | Ground |
|  |  |  |
| BLDC Motor Connections | Three Blue Wires | BLDC ESC connections |

## Control Signal:

30A BLDC ESC requires standard 50-60Hz PWM signal from any remote control as throttle input. You can also generate similar input signal from the microcontroller for making your own customized flying platform. Throttle speed is proportional to the width of the pulse. Maximum throttle position is user programmable. In general throttle is set at zero for 1mS pulse width and full at the 2mS pulse width.

## 3.3.11.5 Alert Tones:

### Abnormal Input Voltage:

ESC checks for the battery voltage at the start-up. If the battery voltage is not in the acceptable range, then ESC starts giving alert signal such with the interval of 1 second.

### Abnormal throttle signal:

When ESC does not detect the throttle signal or it detects abnormality in the throttle signal then it gives sound with the interval of 1 second.

### Throttle stick is not in the zero position:

When throttle stick is not in the zero position at start-up then it will give rapid alert signal with the intervals of 0.25 seconds.

## Battery

An automotive battery or car battery is a rechargeable battery that is used to start a motor vehicle. Its main purpose is to provide an electric current to the electric-powered starting motor, which in turn starts the chemically-powered internal combustion engine that propels the vehicle. Once the engine is running, power for the car's electrical systems is still supplied by the battery, with the alternator charging the battery as demands increase or decrease. Early automobiles did now no longer have batteries, as their electric structures have been limited.

A bell changed used in preference to an electric-powered horn, headlights have been gas-powered, and the engine changed commenced with a crank. Car batteries have become extensively used around 1920 as automobiles have become ready with electric-powered starter motors.

The first beginning and charging structures had been designed to be 6-volt and nice-floor structures, with the car's chassis at once related to the nice battery terminal. Today, nearly all avenue cars have a poor floor machine. The poor battery terminal is hooked up to the car's chassis.

The Hudson Motor Car Company become the primary to apply a standardized battery in 1918 when they commenced the use of Battery Council International batteries. BCI is the organization that units dimensional requirements for batteries.

Cars used 6 V electric structures and batteries till the mid-1950s. The changeover from 6 to 12 V took place while larger engines with better compression ratios required greater electric energy to begin. Smaller cars, which required less energy to begin stayed with 6 V longer, for example, the Volkswagen Beetle in the mid-Nineteen Sixties and the Citroen 2CV in 1970. In the 1990s a 42V electric machine was fashionable to become proposed. It supposed to permit greater effective electrically pushed accessories, and lighter vehicle wiring harnesses.

The availability of better-performance motors, new wiring techniques, virtual controls, and a focal point on hybrid car structures that use high-voltage starters/turbines have in large part removed the frenzy for switching the principal car voltages.

### Yuasa Technical Data Sheet

#### 3.3.5.1.1 Performance

* Voltage 12V
* Capacity (10-hour) 6Ah
* Capacity (20-hour) 6.3Ah
* CCA @ -18°C 100A

3.3.5.1.2 Dimensions (±2mm)

* Length 115mm
* Width 72mm
* Height 132mm

#### 3.3.5.1.3 Weights & Measures

* Mean Weight with Acid 2.5kg
* Acid Volume 0.33l

#### 3.3.5.1.4 Technology

* Technology Cast Ca/Ca
* Separator AGM

Figure 7.20

**Simulink model**

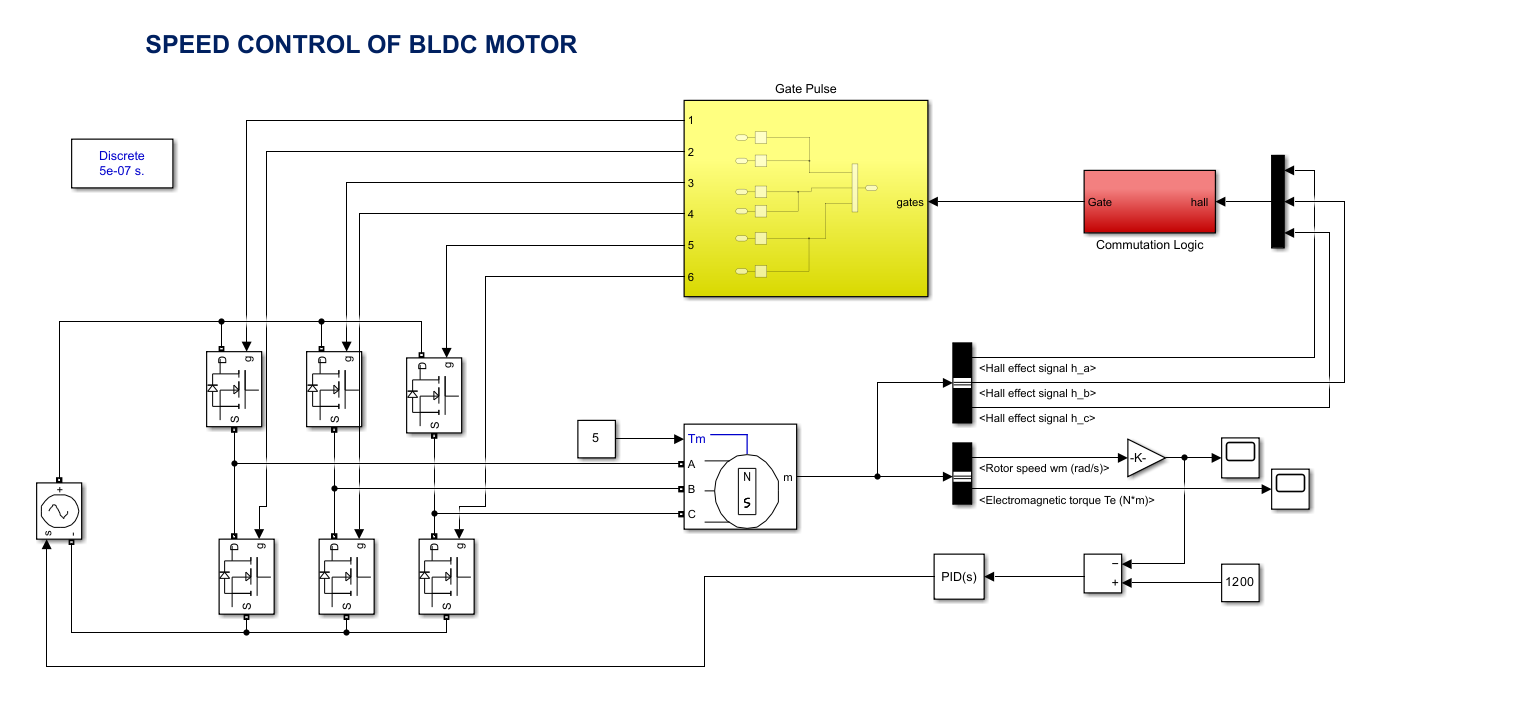


Figure 7.21

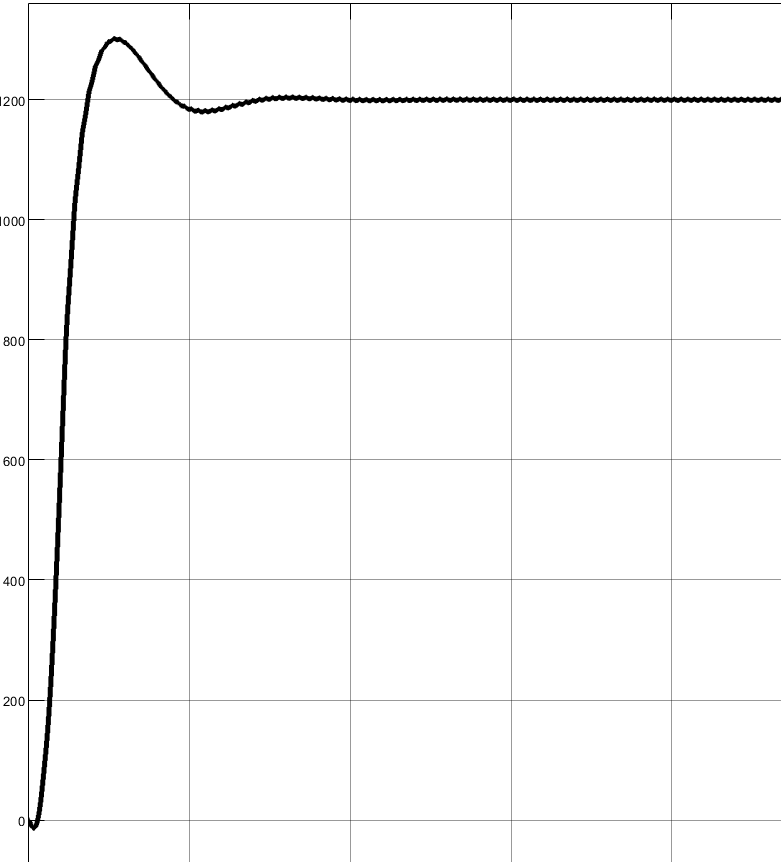


Figure 7.22

# References.

1. Towards Data Science. (n.d.). TensorFlow Lite. Available at: <https://towardsdatascience.com/tensorflow-lite> (Accessed: 24 January 2024).
2. Medium. (n.d.). TensorFlow Lite Optimization. Available at: <https://medium.com/tensorflow/tensorflow-lite-optimization> (Accessed: 24 January 2024).
3. OpenCV. (n.d.). OpenCV Tutorials. Available at: <https://opencv-python-tutroals.readthedocs.io/> (Accessed: 2 Feb 2024).
4. TensorFlow. (n.d.). TensorFlow YouTube Channel. Available at: <https://www.youtube.com/tensorflow> (Accessed: 24 January 2024).
5. Rosebrock, A. (Year of publication). Deep Learning for Computer Vision with Python. PyImageSearch. Available at: <https://www.pyimagesearch.com/deep-learning-computer-vision-python-book/> (Accessed: 24 Feb 2024).
6. NIT Education. (2021). DC Motor. Available at: <https://nit-edu.org/wp-content/uploads/2021/09/ch-29-Dc-motor.pdf> (Accessed: 1 March2024).
7. University of Anbar. (n.d.). DC Motor. Available at: <https://www.uoanbar.edu.iq/eStoreImages/Bank/1939.pdf> (Accessed: 2 March 2024).
8. Google Books. (n.d.). Permanent Magnet Synchronous and Brushless DC Motor Drives. Available at: <https://www.google.com.eg/books/edition/Permanent_Magnet_Synchronous_and_Brushle/U-RxikH3aXEC?hl=ar&gbpv=1&dq=dc+motor+pdf&printsec=frontcover> (Accessed: 24 March 2024).
9. Google Books. (n.d.). Fundamentals of Electrical Drives. Available at: <https://www.google.com.eg/books/edition/Fundamentals_of_Electrical_Drives/2NsGKpLolsQC?hl=ar&gbpv=1&dq=dc+motor+pdf&printsec=frontcover> (Accessed: 27 March 2024).
10. MathWorks. (n.d.). MathWorks Help. Available at: <http://www.mathworks.com/help/index.html> (Accessed: 29 March 2024).
11. Golnaraghi, F., & Kuo, B. C. (Year of publication). Automatic Control Systems. 9th ed. [Publisher].
12. Nise, N. S. (Year of publication). Control Systems Engineering. 6th ed. [Publisher].
13. Dorf, R. C., & Bishop, R. H. (Year of publication). Modern Control Systems. 12th ed. [Publisher].
14. Murray, R. M. (2002). Analysis and Design of Feedback Systems. Caltech. Available at: <https://www.cds.caltech.edu/~murray/courses/cds101/fa02/caltech/astrom-ch6.pdf> (Accessed: 2 April 2024).
15. PDHonline. (n.d.). Control Systems. Available at: <https://pdhonline.com/courses/e331/e331content.pdf> (Accessed: 2 April 2024).
16. Google Search. (n.d.). PID Control. Available at: <https://www.google.com/search?q=pid+pdf&oq=pi&gs_lcrp=EgZjaHJvbWUqCAgAEEUYJxg7MggIABBFGCcYOzIGCAEQRRhAMgYIAhBFGDkyBggDEEUYOzIGCAQQBRhAMgYIBRBFGDwyBggGEEUYPDIGCAcQRRg80gEHOTkwajBqN6gCALACAA&sourceid=chrome&ie=UTF-8> (Accessed: 5 April 2024).
17. Google Books. (n.d.). 100 Technical Questions and Answers for Job Interviews. Available at: <https://www.google.com.eg/books/edition/100_technical_questions_and_answers_for/Fr_tDwAAQBAJ?hl=ar&gbpv=1&dq=pid+pdf&pg=PA113&printsec=frontcover> (Accessed: 6 April 2024).
18. Google Books. (n.d.). Practical PID Control. Available at: <https://www.google.com.eg/books/edition/Practical_PID_Control/ymyAY01bEe0C?hl=ar&gbpv=1&dq=pid+pdf&printsec=frontcover> (Accessed: 4 April 2024).
19. Yedamale, P. (2003). Hands-on Workshop: Motor Control Part 4 - Brushless DC (BLDC) Motor Fundamentals. Microchip AN885.
20. Robinson, S. (2006). Drive and Control Electronics Enhance the Brushless Motor’s Advantages. Apex.
21. Arrigo, D. (2001). L6234 Three Phase Motor Driver. ST AN1088.
22. STMicroelectronics. (2007). Sensorless BLDC Motor Control and BEMF Sampling Methods with ST7MC. ST AN1946.
23. Regional Helix. (2016). Paper 39. Available at: <https://repositorium.sdum.uminho.pt/bitstream/1822/53222/1/Regional_Helix_2016_paper_39.pdf> (Accessed: 6 April 2024).
24. Google Books. (n.d.). Journal of Rehabilitation Research and Development. Available at: <https://www.google.com.eg/books/edition/Journal_of_Rehabilitation_Research_and_D/bkw_HKCDq3sC?hl=ar&gbpv=1&dq=stress+analysis+mechanical+wheelchair&pg=PA232&printsec=frontcover> (Accessed: 24 April 2024).
25. Google Books. (n.d.). Journal of Rehabilitation Research Development. Available at: <https://www.google.com.eg/books/edition/Journal_of_Rehabilitation_Research_Devel/WK4rx7xl5CYC?hl=ar&gbpv=1&dq=stress+analysis+mechanical+wheelchair&pg=PA41&printsec=frontcover> (Accessed: 24 April 2024).
26. Amazon. (n.d.). K10 Wireless Microphone. Retrieved from Amazon K10 Wireless Microphone (Accessed: 5 May 2024).
27. Walmart. (n.d.). K10 Wireless Microphone. Retrieved from Walmart K10 Wireless Microphone​ [https://Walmart.com](https://walmart.com) ​​ (Accessed: 5 May 2024).
28. Chikhale, V., Gharat, R., Gogate, S., & Amireddy, R. (2017). Voice Controlled Robotic System Using Arduino Microcontroller. [Publisher] (Accessed: 10 June 2024).
29. Shinde, S., Singh, J., Jivtode, R., & Khare, H. (2023). Smart Voice-Controlled Wheelchair for Physically Disabled People. [Publisher].
30. YouTube. (2020.). How To Install & Set Up Raspberry Pi OS. Available at: <https://www.youtube.com/watch?v=y45hsd2AOpw> (Accessed: 20 April 2024).
31. YouTube. (2022). Build a Speech Recognition System on a Raspberry Pi. Available at: <https://www.youtube.com/watch?v=vEMzN5RgXbw> (Accessed: 24 April 2024).

# 