# 1. INTRODUCTION

## 1.1 OVERVIEW

The main objective of project is the implementation of a "voice-controlled wheelchair for physically disabled person". Nowadays, there are many kinds of motorized wheelchair available in the market, for instance wheelchair that utilizes the analogue joysticks, touch activated switches and LCD, sip and puff switches, chin-controlled switches, head-controlled switches, tongue-touch pad switches, eye gazed switches, predetermined lines and routes. With all the available methods in the ongoing researches, definitely, the daily lifestyle of the disabled people will be improved. The introduction of the motorized wheelchair increases their independency and mobility in performing their daily social life activities.

By having a wheelchair control system they become more independent. The wheelchair control system employs a voice recognition system for triggering and controlling all its movements. By using the system, the users are able to operate the wheelchair by simply speaking to the wheelchair microphone. The basic movement functions includes forward and reverse direction, left and right turns and stop.

Many physically disabled patients cannot move any of the limbs below the neck. Hence manual and even joystick operated wheelchair is not the solution for these patients. So, the development of voice operated wheelchair will solve the query about the mobility of quadraplegic patient and make them independent of mobility.

## 1.2 PROBLEM STATEMENT

There are many people who are born with disabilities and others get disable through accidents; as days go by it makes feel incompetent and would like to be independent; although sometimes it is not possible for them. Therefore there is need to develop system to aid them in their daily life.

People who are paralyzed below their neck cannot operate the wheelchair based on the joystick movement. To avoid these problems, the voice controlled wheelchair can be mechanized. The voice control system needs to respond to voice commands and avoid obstacles that obstruct its way. The chair must responds to basic commands such as: stop, forward, backward, left, and right. The wheel chair must be reliable and assure customers and its surroundings safety.

## 1.3 OBJECTIVES

The objective of this research is to equip the present motorized wheelchair control system with a voice command system. By having this features, disabled people especially with a severe disabilities that is, unable to move their hand or other parts of a body, are able to move their wheelchair around independently. In order to simplify the operations of the motorized wheelchair as to make it easier and simpler for the disabled person to operate, the voice controlled wheelchair can be adopted. With this simplified operation, many disabled people have a chance to use the system with little training on how to use it. This wheelchair must be able to recognize all the fed voice commands and simultaneously work according to the user's command. The problems faced by using joystick based and motorized wheelchair is to be overcome by implementing the voice controlled wheelchair.

## 1.4 ORGANIZATION OF REPORT

In chapter-1, we have the introduction to the project in which it provides an overview of the proposed system "voice controlled wheelchair". In chapter 2- we have the review of the literature which has an account of previous work done and need of the proposed system. In chapter 3- the thorough description of the proposed system is given. It includes the analysis, modeling and designing of the system. The UML diagrams along with their short description are also included in this chapter. Chapter-4 consists of System Implementation and Coding. Chapter-5 consists of testing that is performed on the system Chapter-6 contains conclusion and future scope for Voice controlled wheelchair.

# 2. REVIEW OF LITERATURE

A literature review is an evaluative report of information found in the literature related to your selected area of study. The review should describe, summarize, evaluate and clarify this literature. It should give a theoretical base for the research and help you (the author) determine the nature of your research.

When conducting research, a literature review is an essential part of the project because it covers all previous research done on the topic and sets the platform on which the current research is based. No new research can be taken seriously without first reviewing the previous research done on the topic.

Wheelchairs have been around for hundreds of years, but early wheelchairs were intended only to help a disabled individual move from one point to another. As society progressed and disabled individuals became more integrated, the role of the wheelchair began to change as well. Wheelchairs today are now considered not only a means of transportation but also as a way to allow users to express their individuality. The literature survey was done after knowing and understanding the limitations of the existing manual and the automated wheelchairs. The usage of normal wheelchairs proves to be a costly affair of the needy, as it does help them to move about to various places as per their need but only when someone helps them to do so.  The basic objective was the development of such a wheelchair that would make them independent. There are also many wheelchairs available which have a joystick to facilitate movements. We also observed that such chairs, although beneficial to the needy but do not always come handy as there could be patients with deformed limbs who cannot possibly operate the joystick with finesse.

Thus it was found that wheelchairs which operate on the voice commands given by the user could prove to be of huge benefit as it would not require them to use any of their limbs but only speech. For patients with speaking disabilities, simpler syllables can be stored to give instructions to the chair. In a better voice recognition system, noise elimination can be a key factor of the operation of the wheelchair smoothly.

## 2.1 LITERATURE SURVEY

In [1], R. Puviarasi, Mritha Ramalingam, Elanchezhian Chinnavan found that several studies have shown the independent mobility, which includes power wheelchair, manual wheelchair and walker access the benefit to both children and adults. Independent mobility reduces dependence on caregivers and family members and promotes feelings of self-reliance. Impaired mobility often results in decreased opportunities to socialize, which leads to social isolation, anxiety and depression. While the needs of many individuals with disabilities can be satisfied with traditional manual or power wheelchair, a segment of the disabled community finds it difficult or impossible to use wheelchairs independently. This population includes individuals with low vision, cognitive deficits, etc.

The idea of using voice-controlled motion for wheelchair is to prove that it can be a unique concept that would stand apart from rest of the manual wheelchairs. The use of this new technology in conjunction with a mechanical system in order to simplify everyday life would spark interest in an ever growing modern society. Many people with disabilities do not have the dexterity necessary to control a joystick on an electrical wheelchair. This can be great for quadriplegics who are permanently unable to move any of their arms or legs. They will be able to move their wheelchair easily by only using voice commands. [2], [3]

In [3], The first known application for controlling a wheelchair was published by Staton et al. The voice interface provided basic commands for wheelchair movement and no real world experiences were performed. In 2002 the RoboChair was developed with three operating modes: Intelligent obstacle avoidance, collision detection and contour following. It uses mixed voice and joystick inputs and a fuzzy control system. A third party speech recognition system was used. The VOIC is a voice operated intelligent wheelchair. The voice recognition uses a neural network for pattern detection with a self-organizing architecture. The system was evaluated on two scenarios, using distinct users and signal to noise ratio. The announced error rate 4 for 5 words vocabulary was 1.8% for a quiet environment and 6.4% with heavy background noise.

Research from University of Notre Dame, 2000, suggests that the current power wheelchair control interfaces used may not, be adequate to provide truly independent mobility for substantial number of person with disabilities. The Respondents to the survey reported on average that approximately ten percent of the patients trained to operate a power wheelchair cannot use the chair upon completion of training for activities of daily living or can do so only with extreme difficulty. [4]

Voice Recognition Kit is being used to recognize the voice command. The voice command given is converted to binary format by Voice Recognition Kit and those binary data is given to the Arduino board for the control of the wheel chair. For example when the user says “forward‟ than chair will move in Forward direction and when he says “Backward‟ than the chair will move in backward direction and similarly for left, right and stop and will be used to detect the command given by the user and will convert into binary form to be sent to the Arduino. [5]

Speech recognition is divided into two broad processing categories; speaker dependent and speaker independent. Speaker dependent systems are trained by the individual that will use the recognizer system. These systems are capable of achieving a high command count and about 98% accuracy for word recognition. Speaker independent is a system that got training to respond to a word regardless of who speaks. Therefore the system must respond to a large number of speech patterns, inflections and enunciations of the target word. Speech recognition also deals with the style of speech it can recognize. There are three styles of speech: isolated, connected and continuous. [6] However, one implementation difficulty is that a voice input system may fail to recognize a user’s voice. Indeed, speech activated interface between human and autonomous/semi-autonomous systems requires accurate detection and recognition. So the pitch and end-point detection plays an important role in speech recognition system [7]

Researches in the area of wheelchair control system are still going on. For an example, a wheelchair controlled by using a tongue. It is design especially for the quadriplegics. This is meant to be used by the disabled person who can only move the body parts above the neck. It utilizes an in-mouth position sensor to control the movements. Another example are a Semi-Autonomous Wheelchair Mobility System (SAWMS) which uses visual tracking technology that utilizes a color camera, sonar, infra-red sensors, contact sensors and a PDA based interface which uses for the wireless communication[8]

There are various hardware alternatives for VR3.

| Sr. No | Name | Features | Price |
|---|---|---|---|
| 1. | HM 2007 | Self-contained stand-alone speech recognition circuit, User programmable, 40 or 20 word vocabulary, Multi-lingual ,Non-volatile memory back up, Easily interfaced to control external circuits & appliances | Rs. 5500 |
| 2. | RSC-4x | Integrates speech-optimized digital and analog processing blocks into a single chip solution capable of accurate speech recognition as well as high quality, low data-rate compressed speech. | $ 2.00 |
| 3. | RKI-1199 | Can recognize 5 commands at one time. Can record up to 15 commands. Working Current <40ma, Small size | Rs. 3800 |

Table 2.1.1 Hardware alternatives to V3

## 2.2 FEASIBILITY STUDY

A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained. “Feasibility Study” is a test of system proposal according to its workability, impact of the organization, ability to meet needs and effective use of the resources. It focuses on these major questions:

What are the user’s demonstrable needs and how does a candidate system meet them?

What resources are available for given candidate system?

What are the likely impacts of the candidate system on the organization?

Whether it is worth to solve the problem?

### 2.2.1 TECHNICAL FEASIBILITY

Technical feasibility is a study of resource availability that may affect the ability to achieve an acceptable system. This evaluation determines whether the technology needed for the proposed system is available or not.

- The hardware system comprised of the Arduino Uno R3, a voice recognition IC, Motors, motor drivers and a battery. It was very important to find a voice recognition IC which is both compatible to the Arduino and is currently available in the market. At the same time it was also essential to find motors which are of low RPM but high stall torque to drive the chairs properly.
- Once all the components are programmed and interfaced well with each other, their physical connections have to be done with extreme care and they need to

be proper. Incorrect connections in any manner would lead to either the system failing to work or any of the components being burnt out.

### 2.2.2 ECONOMICAL FEASIBILITY

Economic justification is generally the “Bottom Line” consideration for most systems. Economic justification includes a broad range of concerns that includes cost benefit analysis. In this we weight the cost and the benefits associated with the candidate system and if it suits the basic purpose of the organization i.e. profit making, the project is making to the analysis and design phase. The financial and the economic questions during the preliminary investigation are verified to estimate the following:

- The hardware comprises of the Arduino Uno R3, Motor drivers L298, Voice recognition module V3, LCD screen (16 x 2) which is economically feasible if compared from an industrial point of view.
- The software comprises using of programming language C.

### 2.2.3 OPERATIONAL FEASIBILITY

It is mainly related to human organizations and political aspects. The following are the operational feasibility:-

- The user has to give the command(s) very clearly. The microphone captures the command, which is recognized by the Arduino. Change in the intonation may result to the command not being recognized by the Voice recognition module.
- The voice recognized is analog signal that is converted into digital signal with the help of Arduino.
- The voice has to be captured in a non noise environment.

### 2.2.4 SCHEDULE FEASIBILITY

Time evaluation is the most important consideration in the development of project. The time schedule required for the developed of this project is very important since more development time effect machine time, cost and cause delay in the development of other systems. The whole system as scheduled was executed and the time slot allotted was feasible to complete the task.

# 3. PROPOSED SYSTEM

## 3.1 DRAWBACKS OF THE CURRENT SYSTEM AND NEED OF THE PROPOSED SYSTEM

In Lever Drive Wheelchair, the rear wheels are driven by the lever through a clutch and bicycle chain. The clutch is engaged by moving the levers inboard, allowing the wheelchair to be driven forward or backward by a similar motion in the lever. Propelling the wheelchair is like rowing a boat. This type of wheel chair cannot be used by physically disabled people as they cannot move the wheelchair manually. Manual wheel chairs or attendant propelled wheel chairs are driven with the help of man power as source of energy for moving the chair, these are self-propelled or propelled with the help of attendee. Wheelchair designed using navigational controls, usually a small joystick mounted on the armrest are not useful for people who cannot move their hands. Other specialist controls may be provided for independent operation of the wheelchair. Some wheelchairs used voice commands like forward, backward, left and right. These commands cannot be given by people with cerebral palsy. A wheelchair where user can automatically reproduce routes taught to system by manually driving wheelchair from starting point to goal point. Uses machine vision to identify landmarks in environment. No obstacle avoidance mode was there.

In the proposed system, we are overcoming the drawbacks of current system by means of using the voice controlled wheelchair. This system is aimed at those who are physically handicapped that is devoid of bipedal motion. A pre-requisite for those paralyzed below the neck, those suffering from quadriplegia, muscular dystrophy or congenital gait abnormalities. There is no need of personal help to push this wheelchair. This system is going to be helpful and used by cerebral palsy patients as well as other type of physically disabled people. Voice commands are given according to the ease of cerebral palsy like patients who are unable to make muscular movements of their jaws so cannot pronounce words like forward, backward, etc.  The motorized wheelchair will increase independency and mobility of physically disabled patients in performing their daily social life activities. Due to the use of voice driven system it's very easy for the people to drive the chair.

## 3.2 PROJECT PLANNING AND SCHEDULING (GANTT CHART)

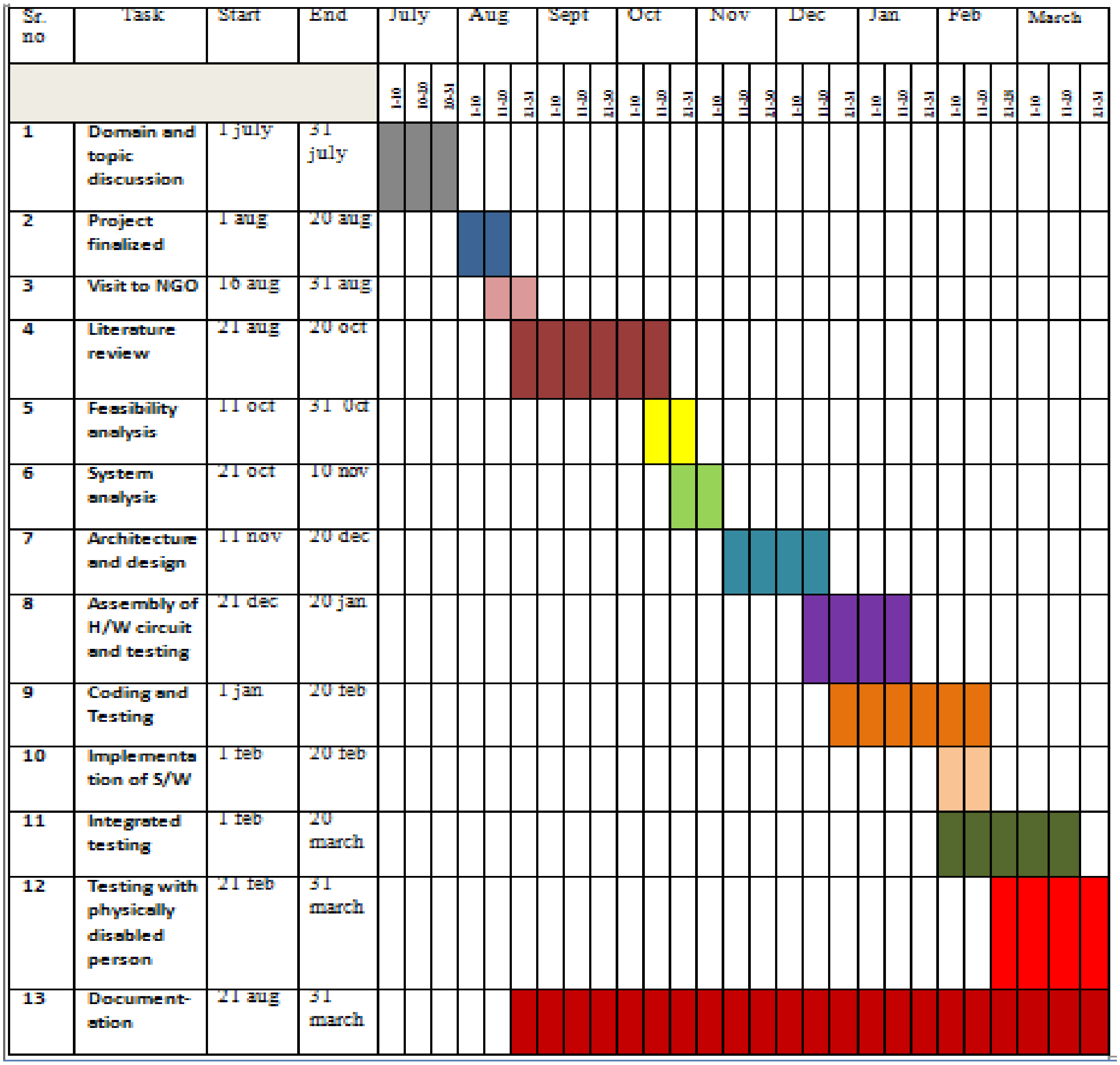
### 3.2.1 PROJECT PLANNING

Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment. Every project needs a road map with clearly defined goals that should not change after the first phase of the project has been completed.

Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure. Project planning is often used to organize different areas of a project, including project plans, workloads and the management of teams and individuals. The logical dependencies between tasks are defined using an activity network diagram that enables identification of the critical path. Project planning is inherently uncertain as it must be done before the project is actually started. Therefore the duration of the tasks is often estimated through a weighted average of optimistic, normal, and pessimistic cases. The critical chain method adds "buffers" in the planning to anticipate potential delays in project execution. Float or slack time in the schedule can be calculated using project management software. Then the necessary resources can be estimated and costs for each activity can be allocated to each resource, giving the total project cost. At this stage, the project schedule may be optimized to achieve the appropriate balance between resource usage and project duration to comply with the project objectives. Once established and agreed, the project schedule becomes what is known as the baseline schedule. Progress will be measured against the baseline schedule throughout the life of the project. Analyzing progress compared to the baseline schedule is known as earned value management.

### 3.2.2 SCHEDULING (GANTT CHART)

A Gantt chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt, an American engineer and social scientist. Frequently used in project management, a Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project.. Following figure represents a Gantt Chart of Voice Control Wheelchair for Physically Disabled Person.



| Sr. no | Task | Start | End |
|---|---|---|---|
| 1 | Domain and topic discussion | 1 july | 31 july |
| 2 | Project finalized | 1 aug | 20 aug |
| 3 | Visit to NGO | 16 aug | 31 aug |
| 4 | Literature review | 21 aug | 20 oct |
| 5 | Feasibility analysis | 11 oct | 31 Oct |
| 6 | System analysis | 21 oct | 10 nov |
| 7 | Architecture and design | 11 nov | 20 dec |
| 8 | Assembly of H/W circuit and testing | 21 dec | 20 jan |
| 9 | Coding and Testing | 1 jan | 20 feb |
| 10 | Implementation of S/W | 1 feb | 20 feb |
| 11 | Integrated testing | 1 feb | 20 march |
| 12 | Testing with physically disabled person | 21 feb | 31 march |
| 13 | Document-ation | 21 aug | 31 march |

Fig. 3.2.2.1 Schedule of Voice Control Wheelchair for Physically Disabled Person

## 3.3 SYSTEM DESCRIPTION AND SRS

### 3.3.1 INTRODUCTION

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide.

The SRS document is prepared as per SRS IEEE (Institute of Electrical and Electronics Engineers) Template. This document is followed by three sections viz. project introduction consisting of purpose, scope of project, and glossary. Second section consists of overall description which includes product perspective, product functions, operating environment, and design & implementation constraints. The third section comprises of functional requirement specification.

#### 3.3.1.1. Purpose

The purpose of this document is to present a detailed description of the Voice Controlled Wheelchair System. It explains the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the wheelchair users and the developers of the system.

#### 3.3.1.2. Scope of Project

This voice controlled wheelchair is hardware and software based system for physically disabled people, which will be flexible to accept non-synthetic commands given by the users. This system is designed for the ease of cerebral palsy and quadriplegic patients who cannot even make jaw movements and give synthetic commands to the wheelchair. The current system otherwise is self-propelled. The use of voice recognition module despite of using joystick in the system will meet the highly disabled users and make easy for them to navigate the system without being dependent on other.

More specifically, this system is designed to allow user to individually draw the system rather than depending on other people to navigate the system according to the users requirements. The voice recognition module will recognize the commands and send it to the Arduino where it is converted into machine understandable language. Thus, voice recognition module will facilitate communication between user and the system via commands.

#### 3.3.1.4. Overview of Document

The Overall Description gives an overview of the functionality of the system. It describes the informal requirements of the system and is used to establish a context for the technical requirements specification in the system. This is the requirements work system that formally specifies Voice Controlled Wheelchair System. Various techniques were used to elicit the requirements of user and identified their needs, analyzed and refined them. The objective of this document therefore is to formally describe the system's high level requirements including functional requirements, non-functional requirements and business rules and constraints.

This document provides an overview of the voice controlled wheelchair. This includes a general description of the system, user characteristics, general constraints, and any assumptions for this system. This model demonstrates that the wheelchair will be simply navigated by the individual user by giving voice commands.

## 3.3.2 OVERALL DESCRIPTION

### 3.3.2.1 Product Perspective

This document contains the problem statement that the current system is facing which is hampering the efficiency of a district cooling plant. It further contains a list of the uses and an admin of the proposed system. It also illustrates the needs and wants of the admin and users that were identified in the brainstorming exercise as a part of the requirements workshops. It further lists and briefly describes the major features and a brief description of each of the proposed system.

### 3.3.2.2 Product Functions

The following SRS contains the details, how our system works from start to end, how the voice recognition module recognizes the voice commands, later how that voice commands are converted into instructions understandable for system, how these instructions drive the motors.  Here we basically convert the voice commands given by the user into system instructions or commands. The system using Arduino for conversion of these voice commands. This system consists of functions which are voice recognition module, Arduino to take voice commands from voice recognition module, and LCD for displaying the commands given by the user.

### 3.3.2.3 Operating Environment

This system uses Arduino for converting the voice commands. The Arduino kit is trained at the first stage to load the commands into it. There are many registers available in Arduino to store the commands. There are five basic commands stored in registers. They are- go, left, right, back and stop. As soon as the voice commands are given to the system, the same will be displayed on the LCD screen. Arduino, voice recognition module and LCD are connected to battery, which is in turn connected to motor drivers. This system can be used in homes, shopping malls, supermarkets, hospitals. The temperature in these places is usually air-conditioned and there is no extreme temperature range to take into account. This chair also operates in parks and streets. Hot days do not act any threaten to the wheel chair design.

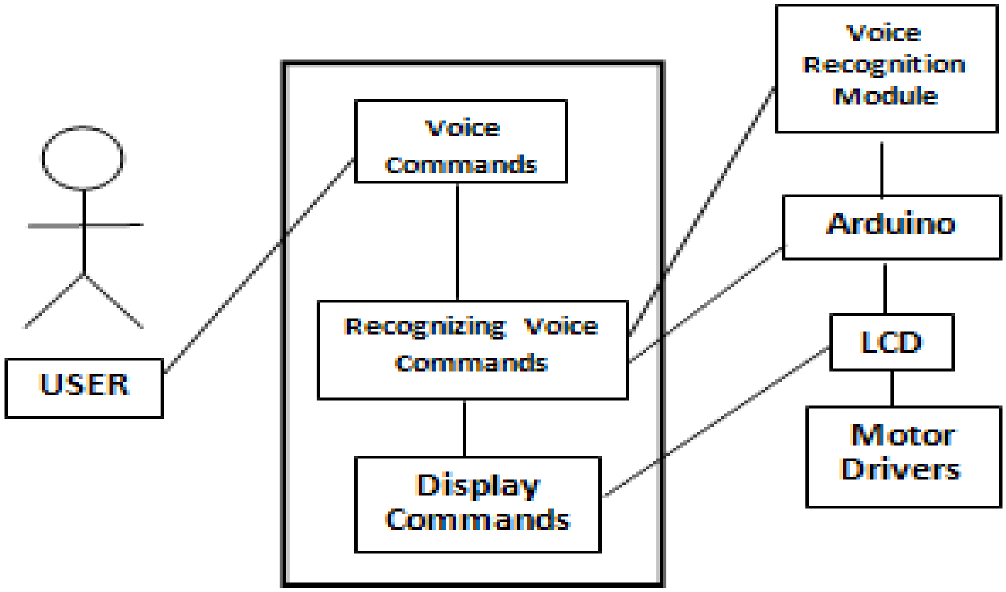



Fig. 3.3.2.3 Operating Environment of Voice Controlled Wheelchair

**3.3.2.4 Design and Implementation Constraints**

The voice commands from the user are taken by mic which is recognized by voice recognition module v3. The voice commands are then fed to Arduino for further processing. Before using the Arduino kit into the system, it should be already trained with the commands which are going to be used by the system user. Arduino uses C programming to store the voice commands into the registers available in it. The commands stored in registers can be change or modify. Each time the voice commands will be given to the system, it will be displayed in the LCD screen as well.

## 3.3.3 FUNCTIONAL REQUIREMENT SPECIFICATION

This section outlines the control flow diagram for the proposed system. The functional requirement of the system will be only voice recognition module and it will not work with the software interface. The system requires an interface between it and the user. This system uses mic as an interface between the both. The mic receives the voice and transfers it to the voice recognition module. This module recognizes the voice commands and transfers these commands to Arduino. If the given commands match the data stored in it, then the commands will be executed.

### 3.3.3.1 Voice Commands Loader Use Case

Use case: Voice Commands Loader Diagram:

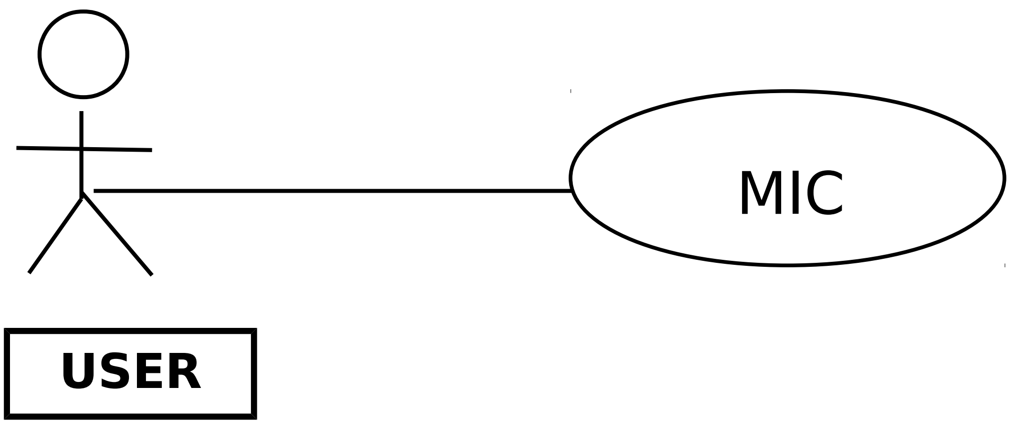



Fig. 3.3.3.1 Voice Commands Loader Use Case Diagram

Brief Description:

The Mic which is connected to wheelchair is used to take the voice commands from the user which is in turn connected with the voice recognition module. The mic take voice commands and pass them to the voice recognition module.

Initial Step-By-Step Description:

1. Locate the mic to its respective place at wheelchair from where it will be convenient for user to give commands.

2. Connect it with the voice recognition module.

3. The mic will pass the commands to the voice recognition module.

### 3.3.3.2 Voice Recognition Use Case

Use case: Voice Recognition Diagram:

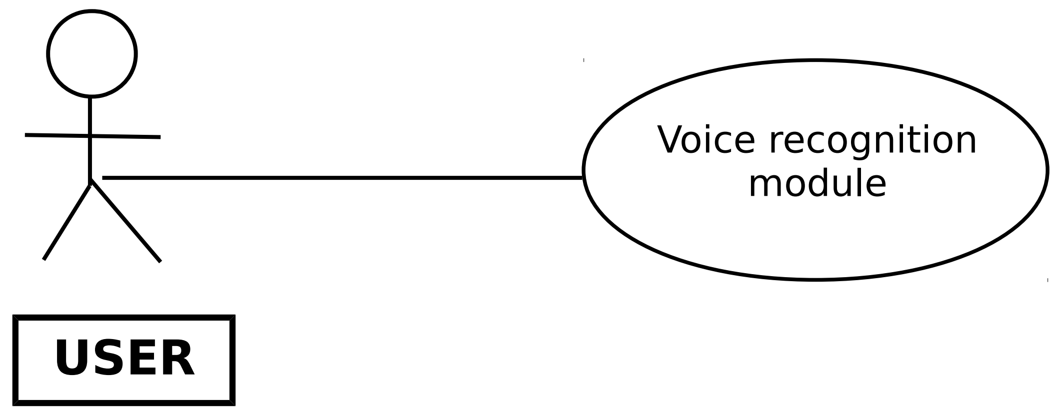



Fig. 3.3.3.2 Voice Recognition Use Case Diagram

Brief Description:

The voice commands given by the wheelchair user is recognized by voice recognition module. It is connected with mic. The mic acts as an interface between the user and the voice recognition module.

### 3.3.3.3 User Use Case

The user has only one use case:

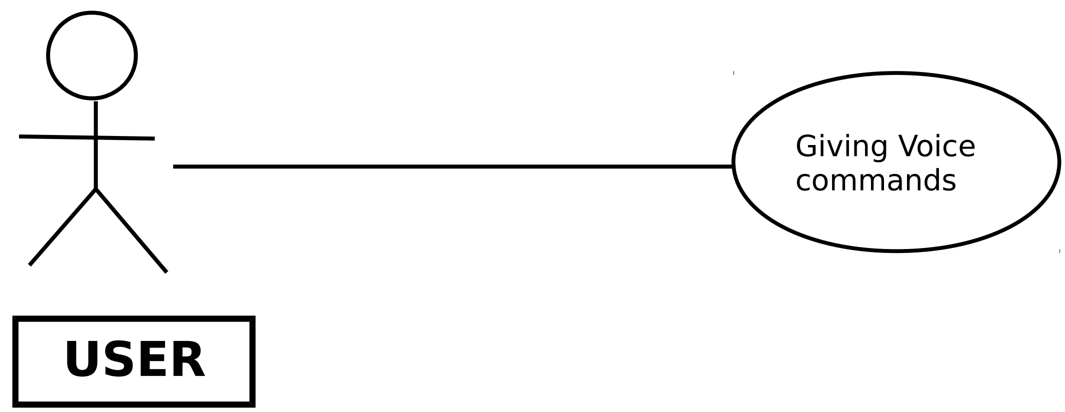



Fig. 3.3.3.3 User Use Case Diagram

#### 3.3.3.4 Voice Recognizing Use Case

Use case: Voice Recognizing Diagram:

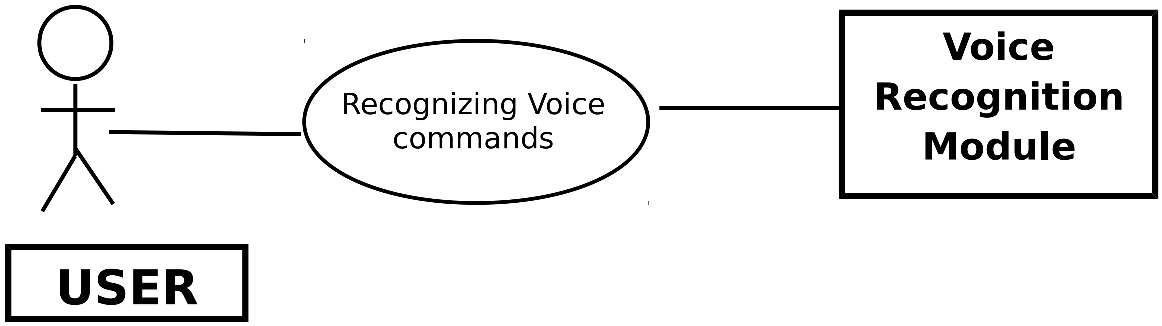



Fig. 3.3.3.4 Voice Recognizing Use Case Diagram

Brief Description:

The user gives voice commands to the system and they are recognized by voice recognition module.

Initial Step-By-Step Description:

1. Give voice commands using which the Arduino kit was trained.

2. Give voice commands correctly.

3. The voice commands are recognized with the help of voice recognition module.

#### 3.3.3.5 Load Recognized Voice Commands Use Case

Use case: load recognized voice commands

Diagram:

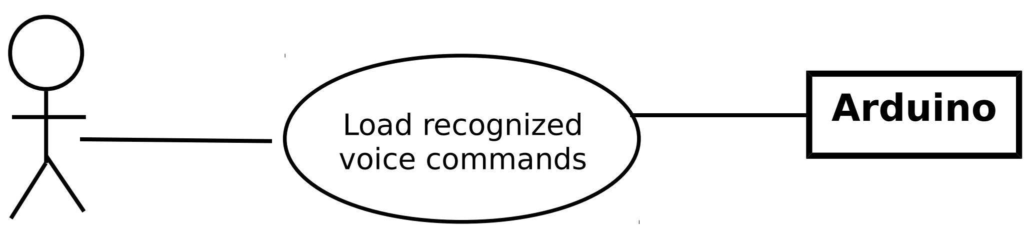



Fig. 3.3.3.5 Load Recognized Voice Commands Use Case Diagram

Brief Description:

The user given commands are load into the Arduino. If it matches then, the instructions will send to the motor drivers and LCD.

Initial Step-By-Step Description:

1. Give voice commands to the Voice recognition module.

2. The commands will be recognized.

3. The Arduino will check these commands with already load commands.

4. If it matches, the motor drivers will receive the instructions and cause movements.

## 3.3.4 REQUIREMENT SPECIFICATION

The only thing required for the system is the voice of the person operating the wheelchair. Apart from that, the voice converter that is voice recognition module is already installed in the system along with the Arduino. For the battery backup of the system, a battery is installed in the chair. To drive the motors of the chair motor drivers is installed in the system, which is interfaced with the Arduino.

### 3.3.4.1 User Characteristics

Users of this voice recognition module hardware based integrated development environment will be both software as well as hardware developers. Since it is reasonable to assume that an average developer has knowledge about functionalities of voice recognition module, the user need not to be informed about the basic functionality of this module as there is no meaning of explaining this module. Only the users need to explain the mic which is the interface between them and this kit as they are going to communicate with this hardware.

### 3.3.4.2 External Interface Requirements

The hardware and software should provide good interface for all the user and the system developer so as to operate on the system, performing the required task such as create, update, viewing the details of the system.

- All the modules provided with the software must fit into this graphical user interface and accomplish to the standard defined.
- The design should be simple and all the different interfaces should follow a standard template.
- The user interface should be able to interact with the user management module and a part of the interface must be dedicated to the start module.

In our system, mic is the user interface using which the user will be able to give commands to the system. The mic is connected to voice recognition module which will then recognize these commands. The user can only provide the commands which will be already fed to the system and if users give commands which is not in the system database then the system will not work.

### 3.3.4.3 Functional Requirements

The functional requirement describes the behavior of the system as it relates to the system's functionality. This system will work only under the hardware module voice recognition module. This module cannot be replaced by any other software. It is able to recognize any type of commands that is both synthetic and non-synthetic. Depending upon the commands fed into the database, it will work.

### 3.3.4.4 Non-Functional Requirements

The proposed system that we developed will be used as independent system for physically disabled people. This system will be used to interact with the highly physically disabled patient as well as people with normal disabilities. Therefore, it is expected that the database would perform functionally all the requirements that are specified by the developer.

- The performance of the system should be fast and accurate.
- Voice Controlled Wheelchair System shall handle expected and non-expected errors in ways that prevent the invalid movement of the wheelchair.
- The system should be able to recognize sound of all frequencies.

## 3.4 SYSTEM ANALYSIS

The system analysis/requirements gathering process is intensive and focused specifically on software. Requirements for both the system and the software are documented and reviewed with the customer. It is a process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning. This involves studying the business process, gathering operational data, understand the information flow, finding out bottlenecks and evolving solutions for overcoming the weaknesses of the system so as to achieve the organizational goals. System analysis also includes dividing the complex process involving the entire system, identification of data store and manual processes.

The major objectives of system analysis are to find answers for each business process; what is being done, how it is being done, who is doing it, when is he doing it, why is it being done and how can it be improved? It is more of thinking process and involves the creative skills of the system analyst. It attempts to give birth to a new efficient system that satisfies the current needs of the user and the scope for future growth within the organizational constraints. The result of this process is the logical system design. System analysis is the iterative process that continues until a preferred and acceptable solution emerges. Requirement analysis also provide software designer with a representation of information, function, and behavior that can be translated to data, architectural, interface, and component-level designs. Finally, the requirements specification provides the developer and the customer with the means to assess quality once software is built. Software requirements analysis may be divided into five areas of effort: (1) Problem recognition, (2) evaluation and synthesis, (3) modeling, (4) specification, and (5) review. Initially, the analyst studies the system specification (if one exists) and the software project plan.

### 3.4.1 FLOWCHART

A flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

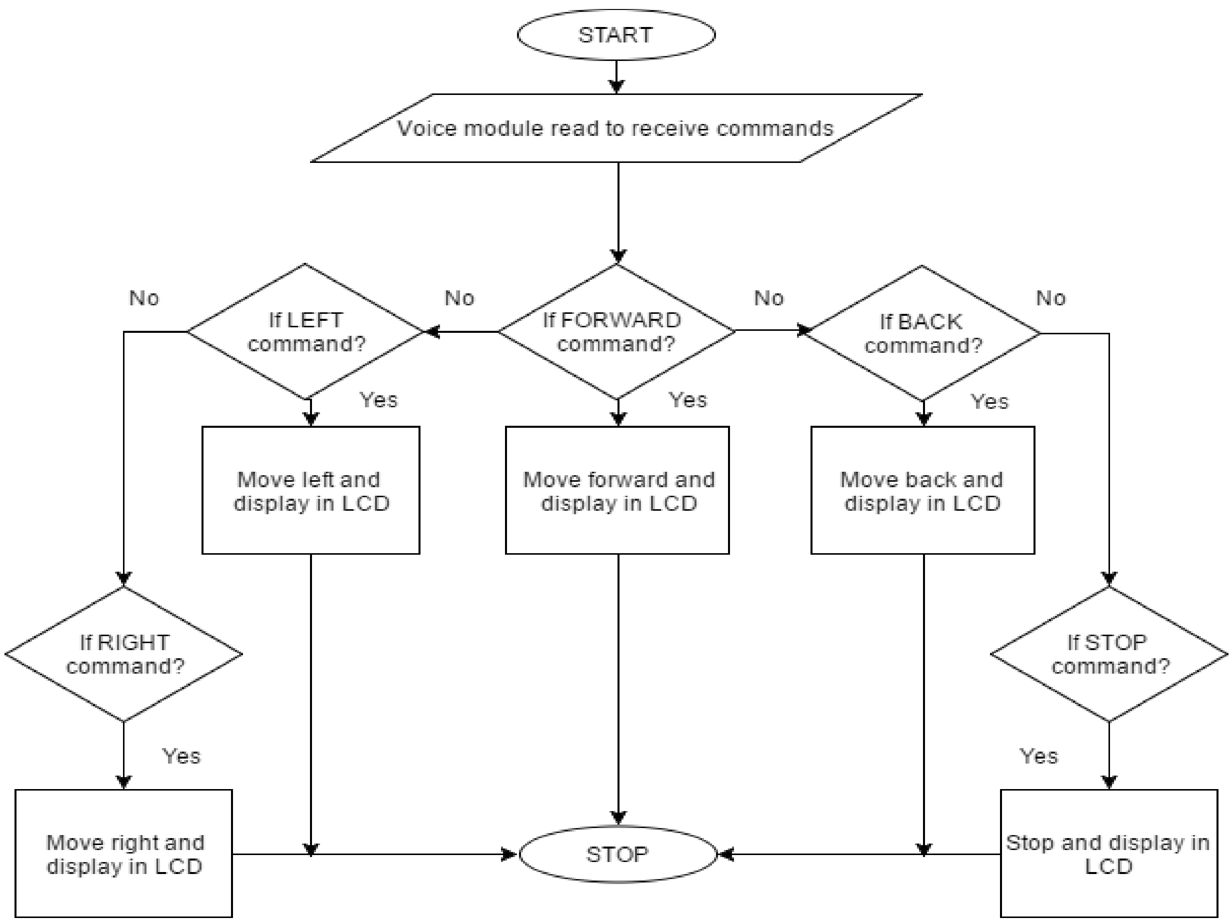



Fig 3.4.1 Flowchart diagram of Voice Control Wheelchair for Physically Disabled Person

### 3.4.2 DFD OF SYSTEM

A data flow diagram is a graphical representation of the ‘flow’ of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored.

The following figure shows the Level 0 DFD of Voice controlled wheel chair for physically disabled person. First the user gives voice input to microphone, then voice is recognized through voice recognition module V3 and gets converted with the help of Arduino Uno R3 and finally the converted voice is sent to motor drivers.


Fig. 3.4.2.1 Level 0 DFD of Voice controlled wheelchair

The next figure shows the Level 1 DFD of Voice controlled wheel chair for physically disabled person. Here the system performs multiple functions as the system interacts with the user. As the diagram depicts, user gives voice input to microphone and the voice recognition module V3 along with Arduino Uno R3 verifies that the given voice input is valid or not. If valid, then the user can access the wheelchair

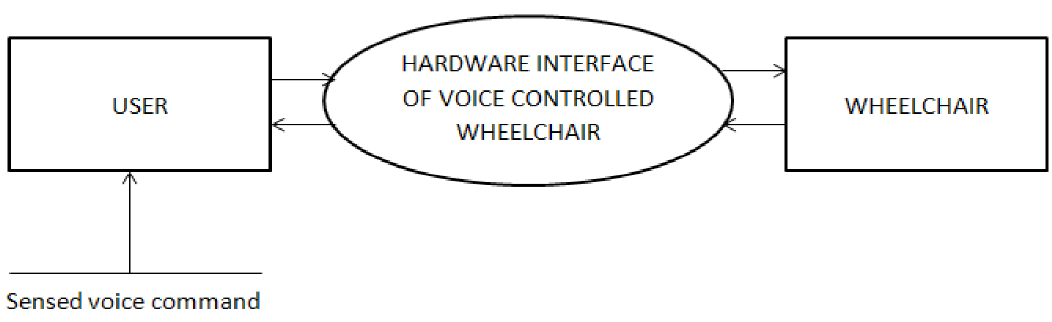



Fig. 3.4.2.2 Level 1 DFD of Voice controlled wheelchair

### 3.4.3 CLASS DIAGRAM

A class diagram in the UML is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code.

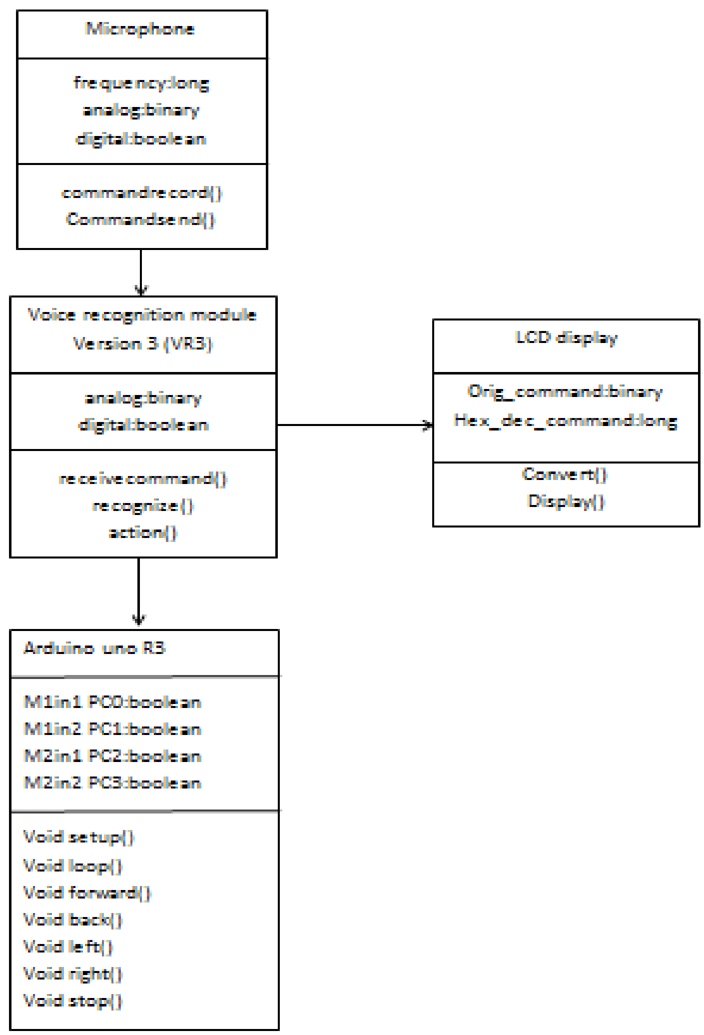



Fig. 3.4.3.1 Class Diagram of Voice controlled wheelchair

### 3.4.4 USE CASE DIAGRAM

A Use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well. Here the users are clients. The module to which admin is having access is shown in the following figure.

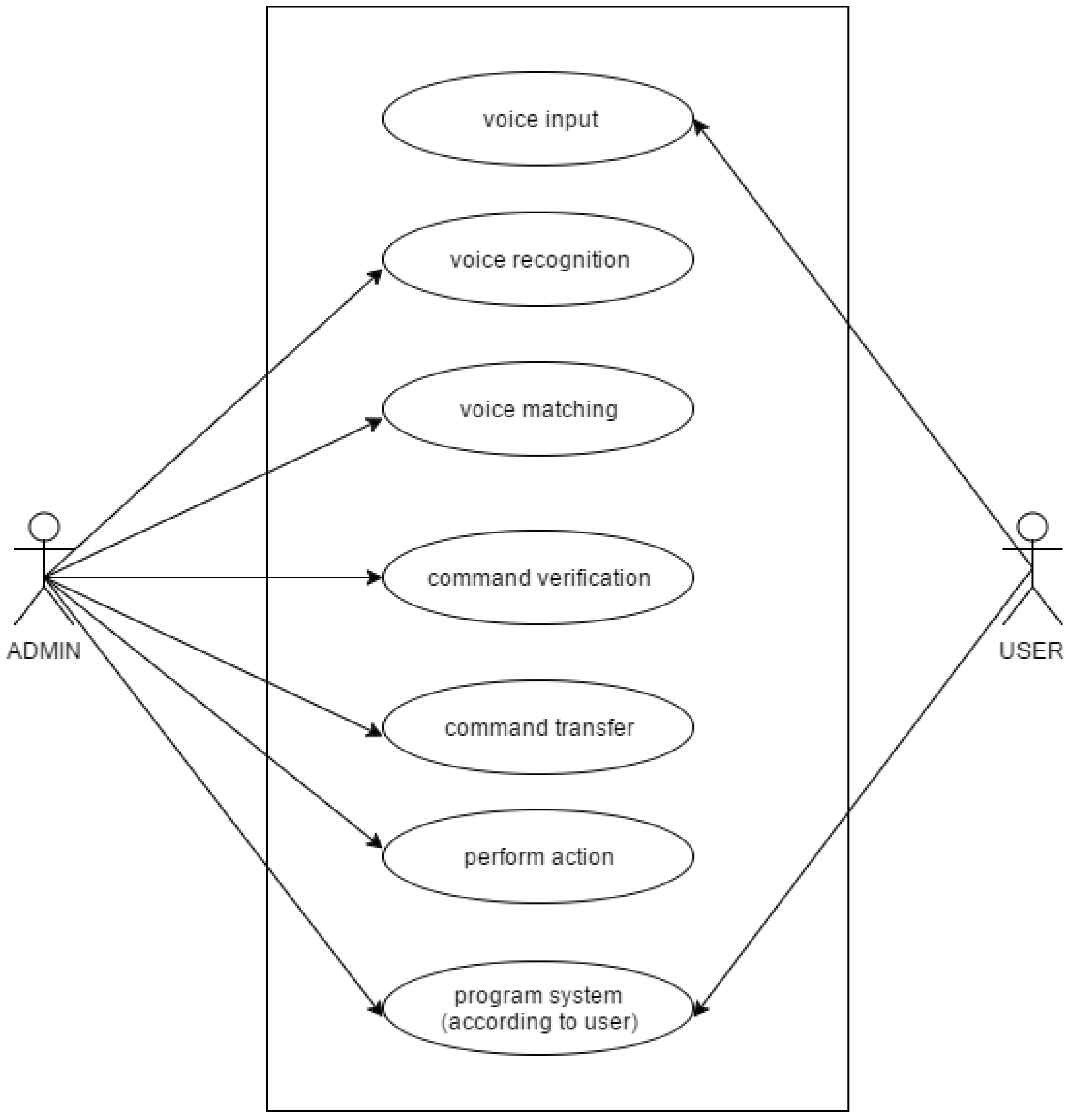



Fig 3.4.4.1 Use Case Diagram for Voice controlled wheelchair

### 3.4.5 COMPONENT DIAGRAM

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems. Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment. The component diagram of voice recognition wheelchair includes the mic, voice recognition module V3, Arduino Uno R3, battery, motor drivers motors and user which interact with each other. After the commands are given by the user, the mic takes it as an input and provides it to the voice recognition module V3, then the voice is converted by the Arduino Uno R3 in the motor accepted form and given to the motor driver and hence the motor moves.

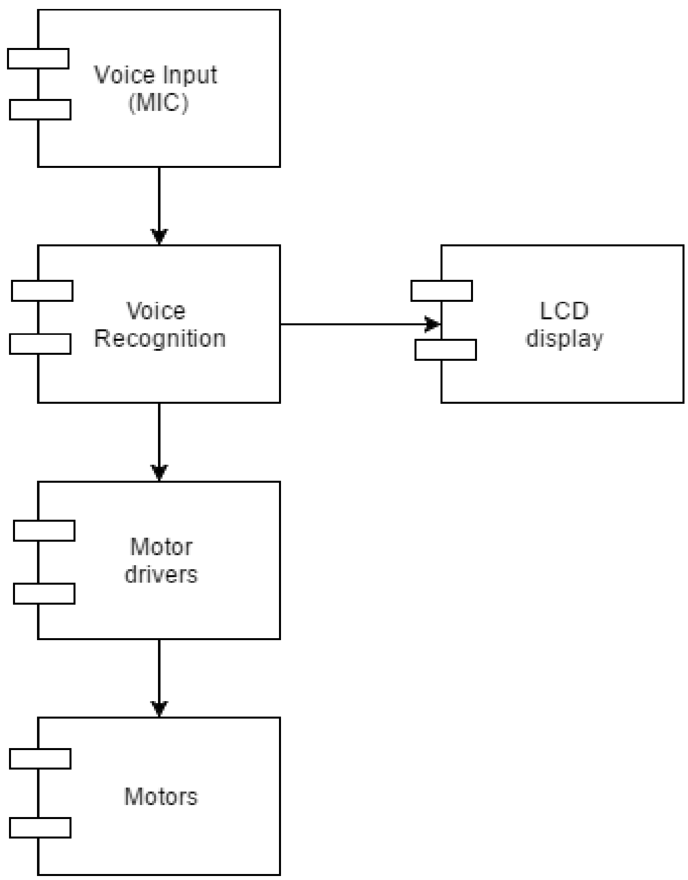



Fig. 3.4.5.1 Component Diagram of voice controlled wheelchair

### 3.4.6 DEPLOYMENT DIAGRAM

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed. So deployment diagrams are used to describe the static deployment view of a system. It consists of nodes and their relationships. The six physical components of voice controlled wheelchair are mic, voice recognition module V3, Arduino Uno R3, motor drivers, motors and user. In this case, the user first gives the voice command to the voice recognition module V3 through mic. Then the Arduino Uno R3 takes this command from the voice recognition module and converts it in the motor accepted form. The motor driver receives the converted voice commands and gives it to motor. Hence the motor moves according to the commands given by the user.

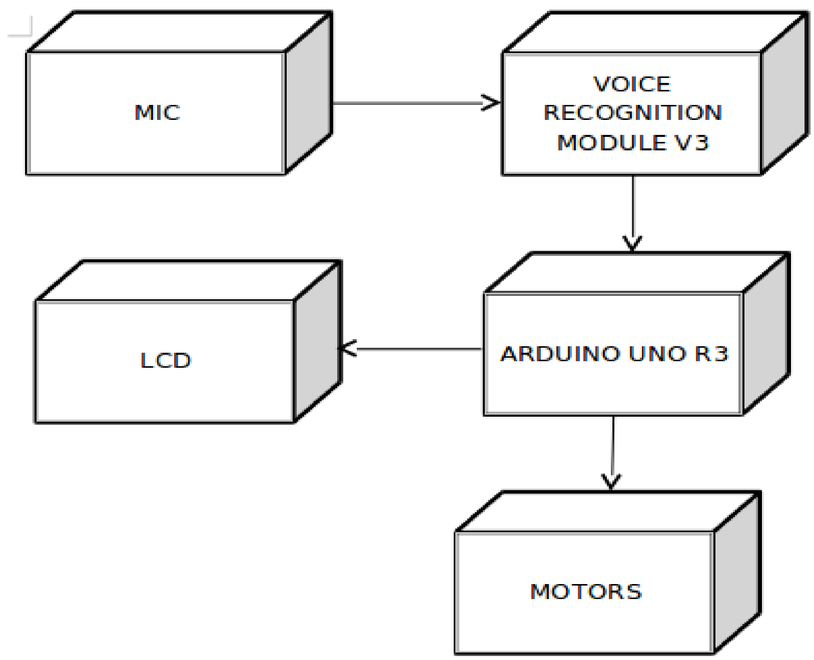



Fig.3.4.6.1 Deployment Diagram of Voice controlled wheelchair

### 3.4.7 HARDWARE AND SOFTWARE REQUIREMENTS

- **Minimum Hardware Requirements**
    1. Arduino Uno R3
    2. L298 DUAL H-bridge motor-controller(2)
    3. 12V DC Magnetized Johnson motor- 100RPM(4)
    4. ELECHOUSE Voice recognition module V3.1
    5. Wheels, 4x10 (4)
    6. LCD board (16x2)
    7. Unilateral microphone
    8. Battery 12v 7.5Ah
    9. Processor: Intel Core i3 and above
- **Minimum Software Requirements**
    1. Arduino nightly 1.8.1 and its libraries
    2. Windows 7 and above

**ARDUINO UNO R3:**

The Arduino Uno board is a microcontroller based on the ATmega328. It has 14 digital input/output pins in which 6 can be used as PWM outputs, a 16 MHz ceramic resonator, an ICSP header, a USB connection, 6 analog inputs, a power jack and a reset button. This contains all the required support needed for microcontroller. In order to get started, they are simply connected to a computer with a USB cable or with a AC-to-DC adapter or battery. Arduino Uno Board varies from all other boards and they will not use the FTDI USB-to-serial driver chip in them. It is featured by the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

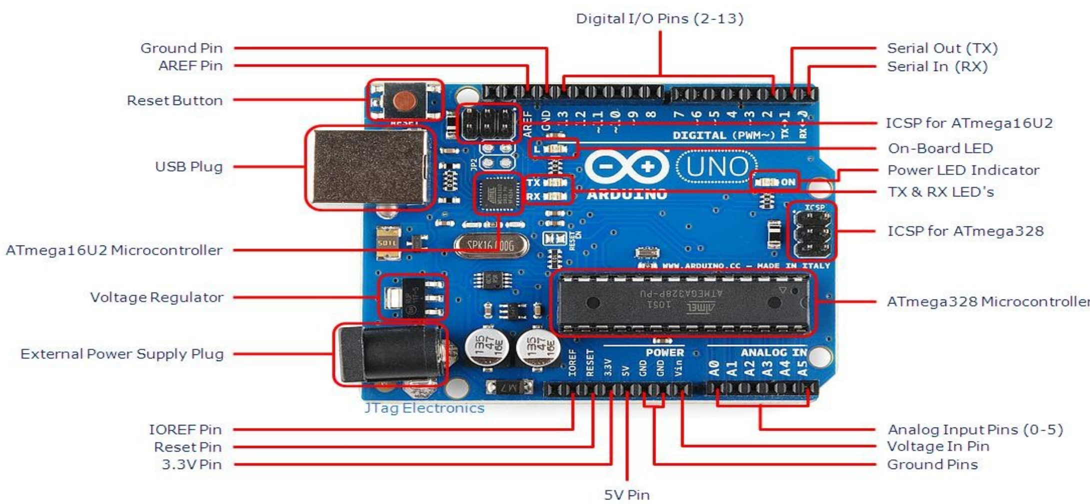



Fig 3.4.7.1 Arduino Uno R3

**L298 DUAL H-BRIDGE MOTOR-CONTROLLER:**

This module is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two motors of up to 2A each in both directions. It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc.

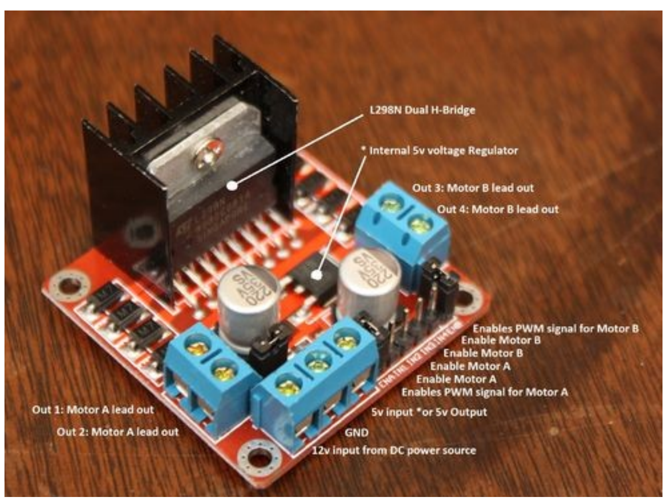



Fig3.4.8.1 L298 motor driver board

**ELECHOUSE VOICE RECOGNITION MODULE V3.1:**

ELECHOUSE Voice Recognition Module is a compact and easy-control speaking recognition board. In V3, voice commands are stored in one large group like a library. Any 7 voice commands in the library could be imported into recognizer. It means 7 commands are effective at the same time.

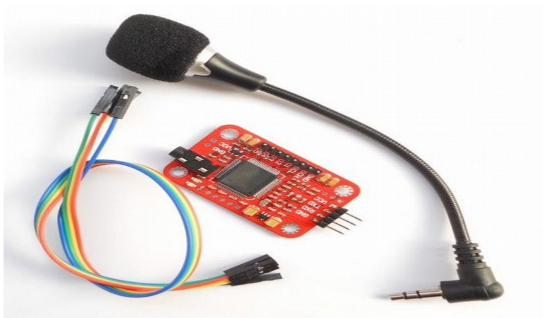


Fig 3.4.9 VR3 board and its components

## 3.5 SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. It is the most crucial phase in the developments of a system. The logical system design arrived at as a result of systems analysis is converted into physical system design.

The system design is of three types which are architectural design, logical design and physical design. The architectural design of a system emphasizes on the design of the systems architecture which describes the structure, behavior, and more views of that system and analysis. The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. The physical design relates to the actual input and output processes of the system.

### 3.5.1 SYSTEM ARCHITECTURE

The system architecture is the structure or structures of the system, which comprise software as well as hardware components, the externally visible properties of those components, and the relationships among them. The architecture is not the operational software. Rather, it is a representation that enables a project to analyze the Effectiveness of the design in meeting its stated requirements, Consider architectural alternatives at a stage when making design change is still relatively easy, and reducing the risks associated with the construction of the software. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. This is defined as a model that helps developers to create flexible and reusable system by breaking it up into different components. As a result, it is likely that if changes are made to a single component, it is possible that the entire system will not need updating when those changes are made.

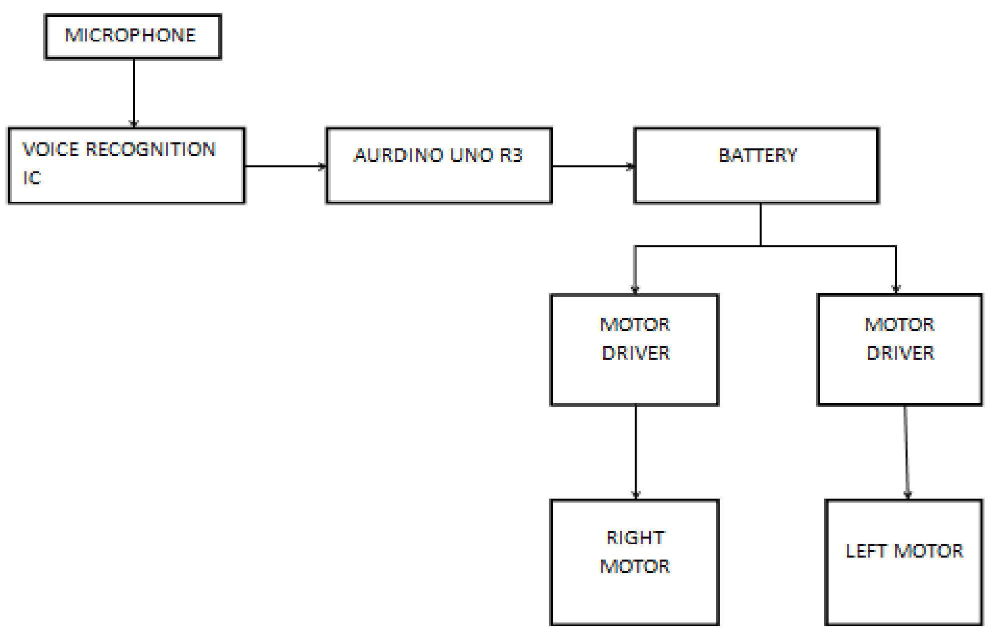



Fig 3.5.1.1 System Block Diagram

Fig.3.5.1.1 shows the system block diagram showing the interconnections between each block and module. This includes a microphone which is located nearest to the user so as to make it handy and easy to use. Generally, the input voice level affects the recognition accuracy result. Principally, the system is triggered by the voice command word produced by the user through the use of this microphone. The user commands for the wheelchair movement by producing words which have been stored previously in the SRAM memory. This SRAM resides in the voice recognition module. Further the processed commands go to the motor driver and thus the motors attached to the wheels move and the movement of the chair presumes. Major five commands are stored in the SRAM: right, left, go, back and stop.

### 3.5.2 WHEELCHAIR DESIGN

#### 3.5.2.1 Front view of the chair

The front view of the chair is basically how the chair looks to a user from front. The components that can be seen from the front is the battery, motor drivers at the bottom. Leg stand is provided for the user for easily sitting on the chair when it moves about. For the ease of keeping the components on place, a wooden plank is fixed and components are placed on it.

Fig 3.5.2.1.1 Front view of the chair

### 3.5.2.2 Back view of the chair

All the major components are fixed at the back of the chair. For the testing of proper functioning of the chair the LCD board is fixed at the back, which displays all the commands given to the wheelchair. In order to get the noise free commands the voice recognition kit is also fixed rather than keeping it free. Since the Arduino is a sensitive module, it's too fixed to avoid any damage to the system's working. Thus, all the connections are fixed by gluing them at the back .

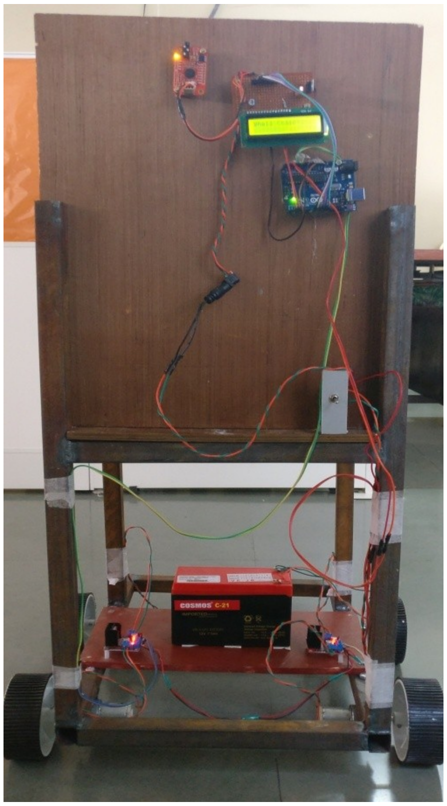


Fig. 3.5.2.2.1 Back view of the chair

# 4. SYSTEM IMPLEMENTATION & CODING

## 4.1 IMPLEMENTATION

The voice controlled wheelchair is built using Arduino Uno. Arduino Uno provides the interface for voice recognition module and motor driving module. The voice recognition module being used for this project is ELECHOUSE Voice recognition module Version 3 (VR3) and the motor drivers used are l298 drivers. The commands for driving the chair are stored in voice recognition IC as well as Arduino.

### 4.1.1 Voice Recognition Module

This module is responsible for taking the input from the user. The user input is in the form of voice commands. The commands that we have selected for our system are: Go, Left, Right, Back and Stop. Our Voice Recognition module would recognize only these 5 commands. The hardware component being used for voice recognition is ELECHOUSE Voice recognition module Version 3.This product is a speaker-dependent voice recognition module. It supports up to 80 voice commands in all. Max 7 voice commands could work at the same time. Any sound could be trained as command. Users need to train the module first before let it recognizing any voice command.

ELECHOUSE Voice Recognition Module is a compact and easy-control speaking recognition board. This board has 2 controlling ways: Serial Port (full function), General Input Pins (part of function). General Output Pins on the board could generate several kinds of waves while corresponding voice command was recognized. In V3, voice commands are stored in one large group like a library. Any 7 voice commands in the library could be imported into recognizer. It means 7 commands are effective at the same time.

Parameters of v3:

- Voltage: 4.5-5.5V
- Current: <40mA
- Digital Interface: 5V TTL level for UART interface and GPIO

- Analog Interface: 3.5mm mono-channel microphone connector + microphone pin interface
- Size: 31mm x 50mm
- Recognition accuracy: 99% (under ideal environment)

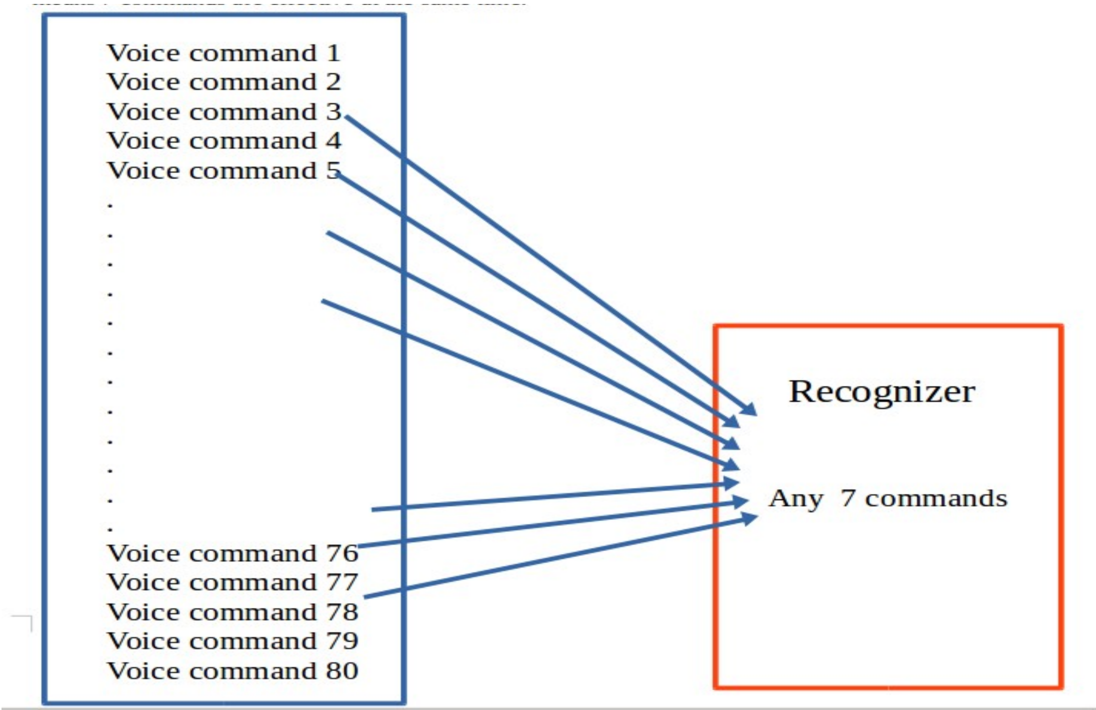



Fig 4.1.1.1 Command storage and recognition

Terminology

- VR3 -- Voice Recognition Module V3
- Recognizer -- a container where acting voice commands (max 7) were loaded. It is core part of voice recognition module. For example, it works like "playing balls". You have 80 players in your team. But you could not let them all play on the court together. The rule only allows 7 players playing on the court. Here the Recognizer is the list which contains names of players working on the court.

- Recognizer index -- max 7 voice commands could be supported in the recognizer. The recognizer has 7 regions for each voice command. One index corresponds to one region: 0~6
- Train -- the process of recording your voice commands
- Load -- copy trained voice to recognizer
- Voice Command Record -- the trained voice command store in flash, number from 0 to 79
- Signature -- text comment for record

The connections between VR3 and Arduino are as follows:

| Arduino | | VR Module |
|---|---|---|
| 5V | ----> | 5V |
| 2 | ----> | TX |
| 3 | ----> | RX |
| GND | ----> | GND |

Fig 4.1.1.2 Arduino and vr3 connections

For training, following steps need to be performed:

1. Open vr_sample_train (File -> Examples -> VoiceRecognitionV3 -> vr_sample_train)
2. Choose right Arduino board (Tool -> Board, UNO), Choose right serial port.
3. Click Upload button, wait until Arduino is uploaded.
4. Open Serial Monitor. Set baud rate 115200, set send with Newline.

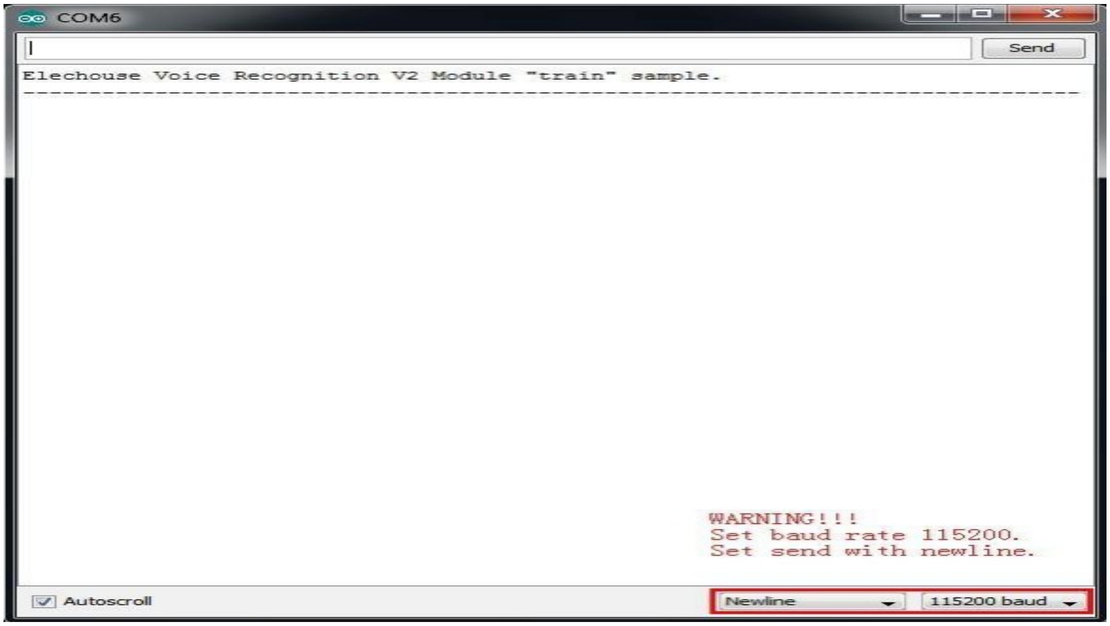



Fig. 4.1.1.3 Arduino serial monitor

5. Send command settings (case insensitive) to check Voice Recognition Module settings. Input settings, and hit Enter to send.

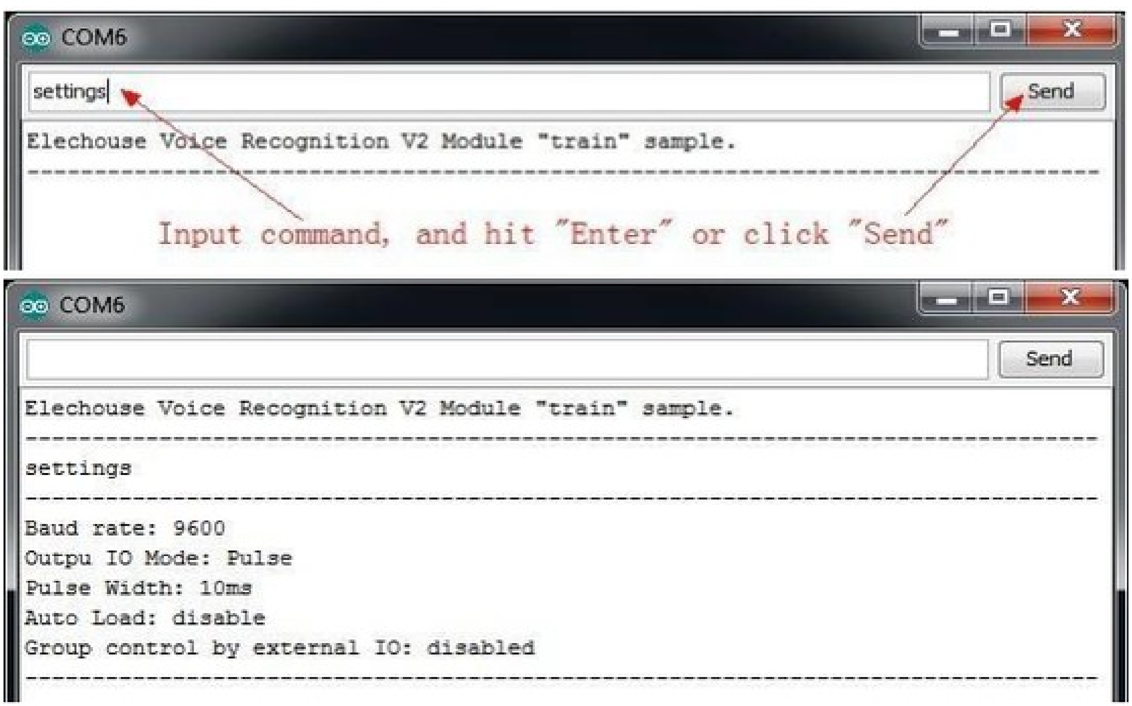



Fig 4.1.1.4 Checking voice recognition settings

6. Train Voice Recognition Module. Send **sigtrain 2 left** commands to train record 2 with signature "left". The records are from 0-6. When Serial Monitor prints "Speak now", you need speak your voice(can be any word, meaningful word recommended, may be 'Left' here), and when Serial Monitor prints "Speak again", you need repeat your voice again. If these two voice samples match, Serial Monitor prints "Success", and "record 2" is trained, or if are not matched, repeat speaking until success. (Signature is a piece of text description for the voice command). The signature could be displayed if its command was called. When training, the two LEDs on the Voice Recognition Module can indicate the training process. After sending the training command, the SYS_LED (yellow) is blinking fast which remind you to get ready. Speak your voice command as soon as the STATUS_LED (red) light lights on. The recording process ends once when the STATUS_LED (red) lights off. Then the SYS_LED is blinking again, get ready for next recording process. When the training process ends successful, SYS_LED and STATUS_LED blink together. If the training fails, SYS_LED and STATUS_LED blink together, but quickly.

```
Signature:l
-------------------------------------------------------------------------------
sigtrain 0 on
-------------------------------------------------------------------------------
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Cann't matched
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Cann't matched
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Success
Success: 1
Record 0        Trained
SIG: on
-------------------------------------------------------------------------------
sigtrain 1 off
-------------------------------------------------------------------------------
Record: 1       Speak now
Record: 1       Speak again
Record: 1       Cann't matched
Record: 1       Speak now
Record: 1       Speak again
Record: 1       Success
Success: 1
Record 1        Trained
SIG: off
-------------------------------------------------------------------------------
```

Fig 4.1.1.5 Voice train

7. Train another record. Send **sigtrain 1 Off** command to train record 1 with signature "Off". Choose your favorite words to train (it can be any word, meaningful word recommended, may be 'Off' here)

8. Send load 01 command to load voice. And say your word to see if the Voice Recognition Module can recognize your words.


Fig 4.1.1.6 Load command

If the voice is recognized, you can see.

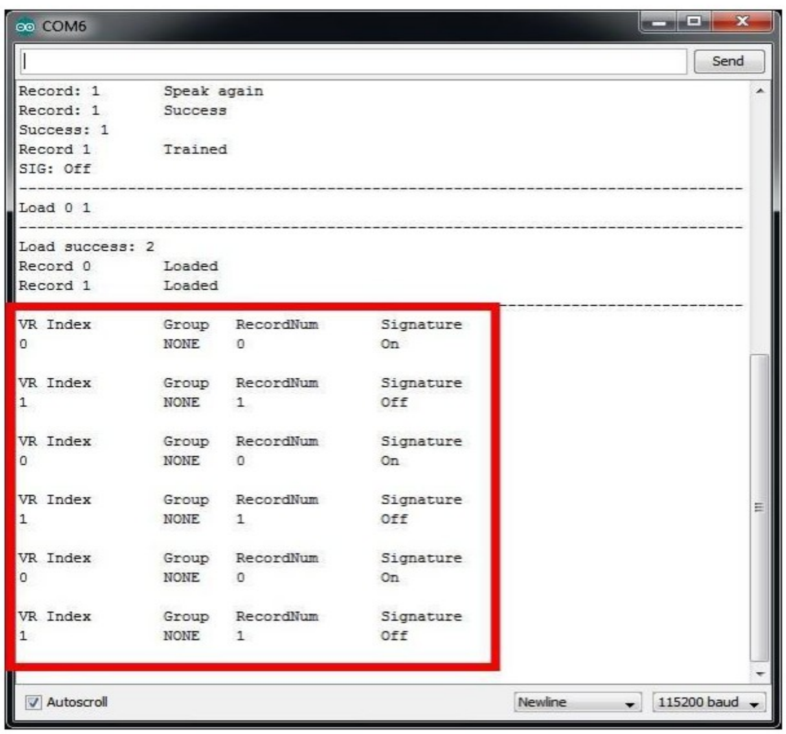



Fig. 4.1.1.7 Recognized commands

9. Train finish. Train sample also support several other commands.

```
-------------------------------------------------------------------------------
train        train (r0) (r1)...          train 0 2 45          Train records
load         load (r0) (r1) ...          load 0 51 2 3         Load records
clear        clear                       clear                 remove all records in  Recognizer
record       record / record (r0) (r1)... record / record 0 79  Check record train status
vr           vr                          vr                    Check recognizer status
getsig       getsig (r)                  getsig 0              Get signature of record (r)
sigtrain     sigtrain (r) (sig)          sigtrain 0 ZERO       Train one record(r) with signature(sig)
settings     settings                    settings              Check current system settings
help         help                        help                  print this message
```

Fig. 4.1.1.8 Other commands

### 4.1.2 Command Display

This module consists of a 16*2 LCD board, that is,16 words and 2 lines display, that takes the input from voice recognition module and the input is converted to hexadecimal value in Arduino. For each command, the corresponding direction of movement of chair is displayed on LCD board. The power supply to LCD is provided via Arduino. It requires power supply of 5V.

The word displayed for each command is as follows:

| **Command** | **LCD Display** |
|---|---|
| GO | FORWARD |
| LEFT | LEFT |
| RIGHT | RIGHT |
| BACK | REVERSE |
| STOP | STOP |


Fig 4.1.2.1 LCD display

### 4.1.3 Motor driving module

This module is the main module of the system. It consists of the motors and motor drivers. This module takes the output from Voice Recognition Module and drives the motor in accordance with the command given by the user. The main hardware component in this module is l298.Motor driver which is a dual bidirectional motor driver based on the very popular L298 Dual HBridge Motor Driver IC. This is well suited for connection to a microcontroller requiring just a couple of control lines per motor. The logic for driving motor is provided in Arduino program. The Arduino acts as an interface between motor driver and the motors.

#### 4.1.3.1 Motors

The motors being used for this system are 12V DC magnetized Johnson motors-100RPM

Rotations per minute: 100

Output Torque Range: 15kg-cm to 20kg-cm

**Specifications:**

10 RPM 12V DC Motor with Gear Box.

Heavy Duty Metal Gears.

Shaft: 6 mm

Torque: 15 kg-cm

No Load Current: 500 MA (MAX),

LOAD CURRENT= 700 MA (MAX).

The system contains 4 motors which are connected to 4 wheels and two motors are connected to two motor drivers.

Fig 4.1.3.1.1 Motors

**4.1.3.2 Motor drivers**

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

The pin diagram for the motor driver IC is as follows:

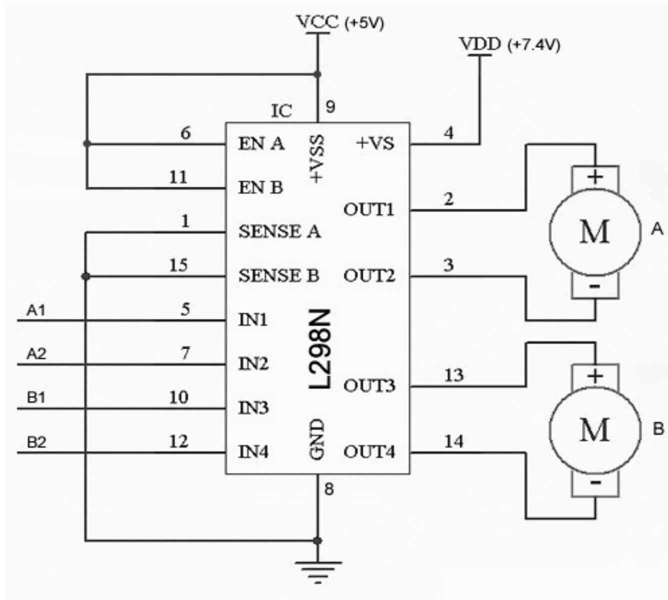



Fig.4.1.3.2.1 L298N pin diagram

Before transferring the commands to motors, they need to be converted into hexadecimal form. The hexadecimal forms for used commands are:

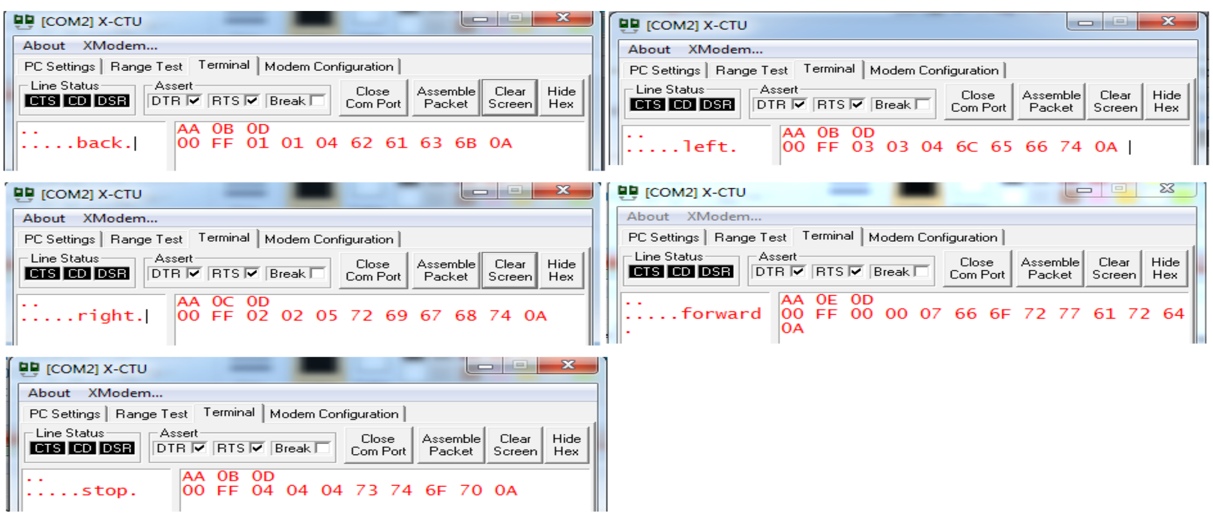



Fig. 4.1.3.2.2 Hexadecimal commands

The system has two motor drivers, as one motor driver can drive 2 motors. The logic required by motor driver to drive motor is programmed via Arduino.

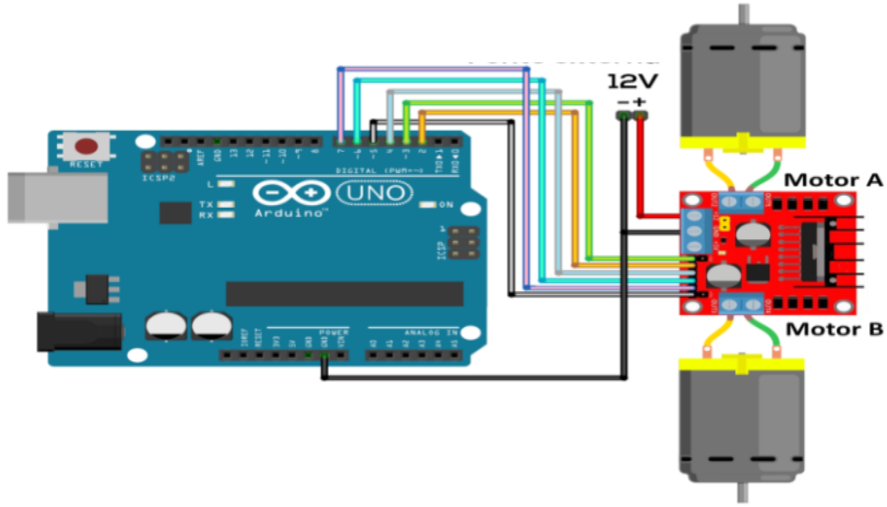



Fig4.1.3.2.3 Motor driver circuit diagram

**4.1.3.3 Wheels:**

The wheels compatible with the motors used have dimensions 4(width) x 10(diameter).

Fig 4.1.3.3.1 Wheels attached to the motor

The actual circuit for motor driving is

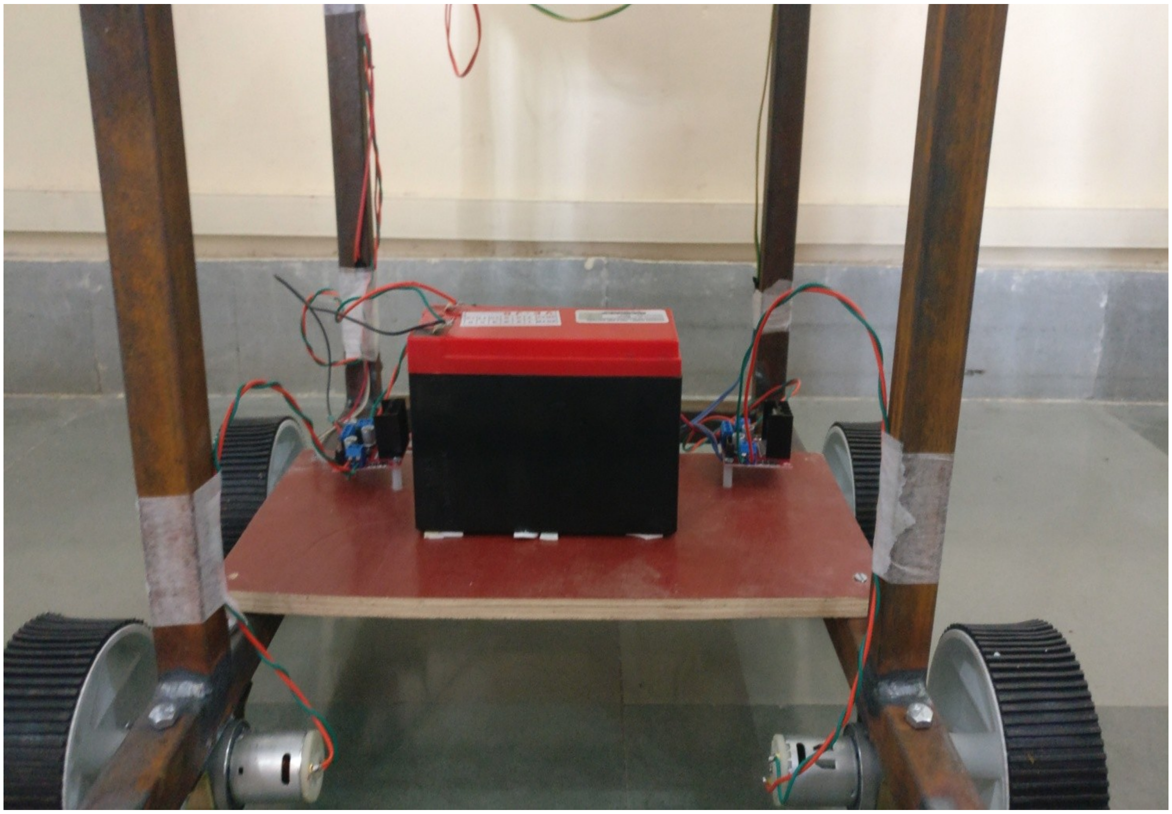


Fig. 4.1.3.3.2 Motor driving circuit

### 4.1.4 Arduino Uno R3

Arduino/Genuino Uno is a microcontroller board based on the ATmega328P . It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller. Arduino, in this wheelchair, is the base of the system. It provides interface to voice recognition module and motor driving module.

### 4.1.4.1 For LCD display

For LCD display, simple print command is used as

*wheelvoice.ino (~/Desktop/Project) - gedit

Open | Save

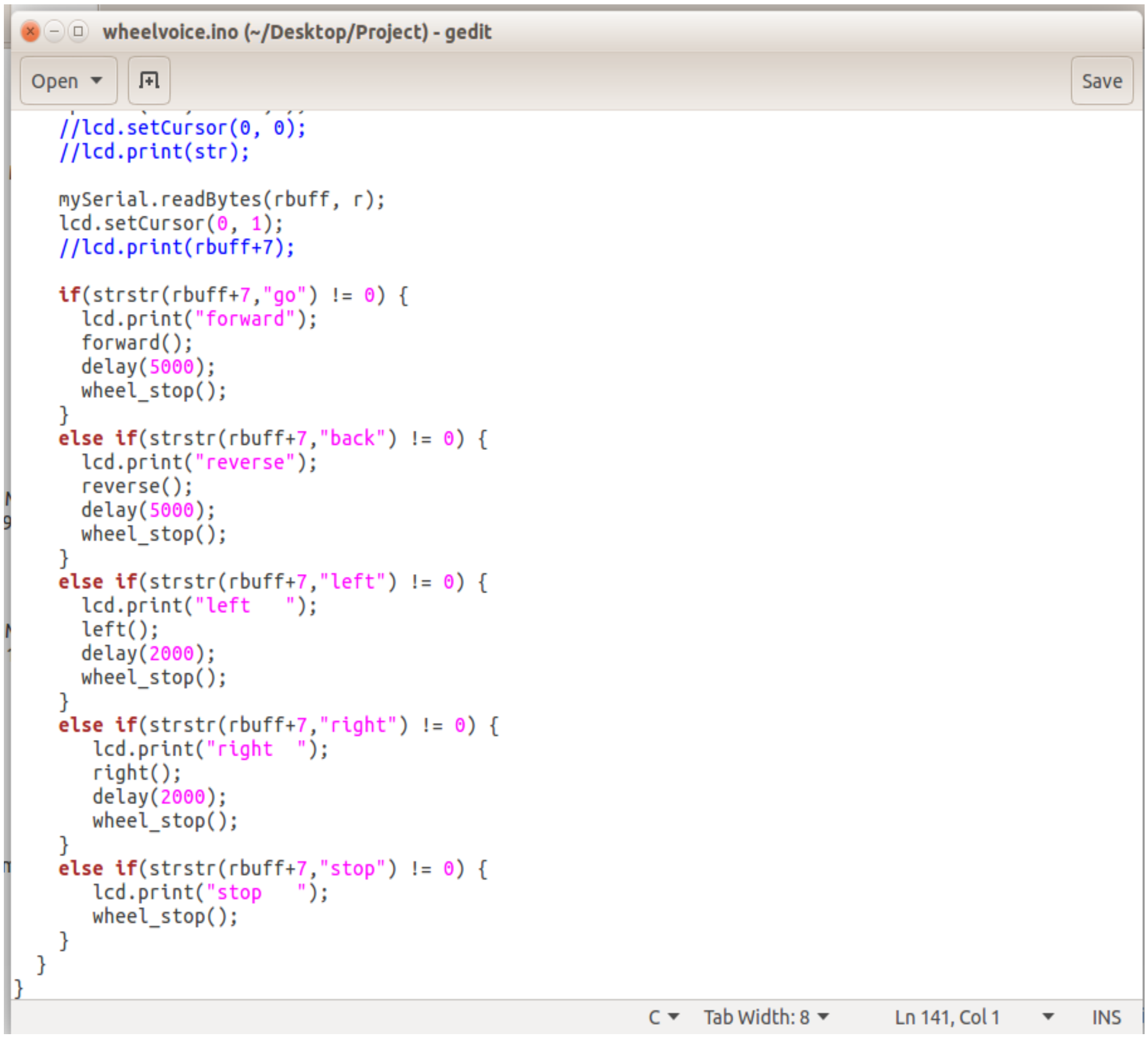
```
            wheel_stop();
            break;
    }
    mySerial.write(ch);
  }*/
  if (mySerial.available()) {
    delay(100);
    int r = mySerial.available();

    sprintf(str,"r:%d",r);
    //lcd.setCursor(0, 0);
    //lcd.print(str);

    mySerial.readBytes(rbuff, r);
    lcd.setCursor(0, 1);
    //lcd.print(rbuff+7);

    if(strstr(rbuff+7,"go") != 0) {
      lcd.print("forward");

    }
    else if(strstr(rbuff+7,"back") != 0) {
      lcd.print("reverse");
      reverse();
      delay(5000);
      wheel_stop();
    }
    else if(strstr(rbuff+7,"left") != 0) {
      lcd.print("left    ");
      left();
      delay(2000);
      wheel_stop();
    }
    else if(strstr(rbuff+7,"right") != 0) {
       lcd.print("right   ");
       right();
       delay(2000);
```

C ▾ Tab Width: 8 ▾ Ln 89, Col 7 ▾ INS

Fig 4.1.4.1.1 LCD display code snippet

### 4.1.4.2 For driving motor

For driving motors, the respective motors are set “high” depending on the command and they stay high for the given delay time. The delay time can be changed depending on the weight imposed on motor. The delay time should be greater for higher loads and lower for small or no load.




```
  //lcd.setCursor(0, 0);
  //lcd.print(str);

  mySerial.readBytes(rbuff, r);
  lcd.setCursor(0, 1);
  //lcd.print(rbuff+7);

  if(strstr(rbuff+7,"go") != 0) {
    lcd.print("forward");
    forward();
    delay(5000);
    wheel_stop();
  }
  else if(strstr(rbuff+7,"back") != 0) {
    lcd.print("reverse");
    reverse();
    delay(5000);
    wheel_stop();
  }
  else if(strstr(rbuff+7,"left") != 0) {
    lcd.print("left    ");
    left();
    delay(2000);
    wheel_stop();
  }
  else if(strstr(rbuff+7,"right") != 0) {
     lcd.print("right   ");
     right();
     delay(2000);
     wheel_stop();
  }
  else if(strstr(rbuff+7,"stop") != 0) {
     lcd.print("stop    ");
     wheel_stop();
  }
 }
}
```

Fig 4.1.4.2.1 Motor driving code snippet

The functions for driving motors are written as:

```
wheelvoice.ino (~/Desktop/Project) - gedit

    else if(strstr(rbuff+7,"right") != 0) {
       lcd.print("right  ");
       right();
       delay(2000);
       wheel_stop();
    }
    else if(strstr(rbuff+7,"stop") != 0) {
       lcd.print("stop   ");
       wheel_stop();
    }
  }
}

void forward(void) {
  motorPORT |= (1 << m1in1 | 1 << m2in1);
  motorPORT &= ~( 1 << m1in2 | 1 << m2in2);
}

void reverse(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m2in1);
  motorPORT |= ( 1 << m1in2 | 1 << m2in2);
}

void left(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m2in2);
  motorPORT |= ( 1 << m1in2 | 1 << m2in1);
}

void right(void) {
  motorPORT |= (1 << m1in1 | 1 << m2in2);
  motorPORT &= ~( 1 << m1in2 | 1 << m2in1);
}

void wheel_stop(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m1in2 | 1 << m2in1 | 1 << m2in2);
}

C   Tab Width: 8   Ln 141, Col 1   INS
```

Fig 4.1.4.2.2 Motor driving logic

Each motor is assigned value as HIGH or LOW depending on the direction of motor. For example, to move in forward direction, all motors are set HIGH and to move in left direction, only the motors on right side are set HIGH.

The actual system implementation looks like:

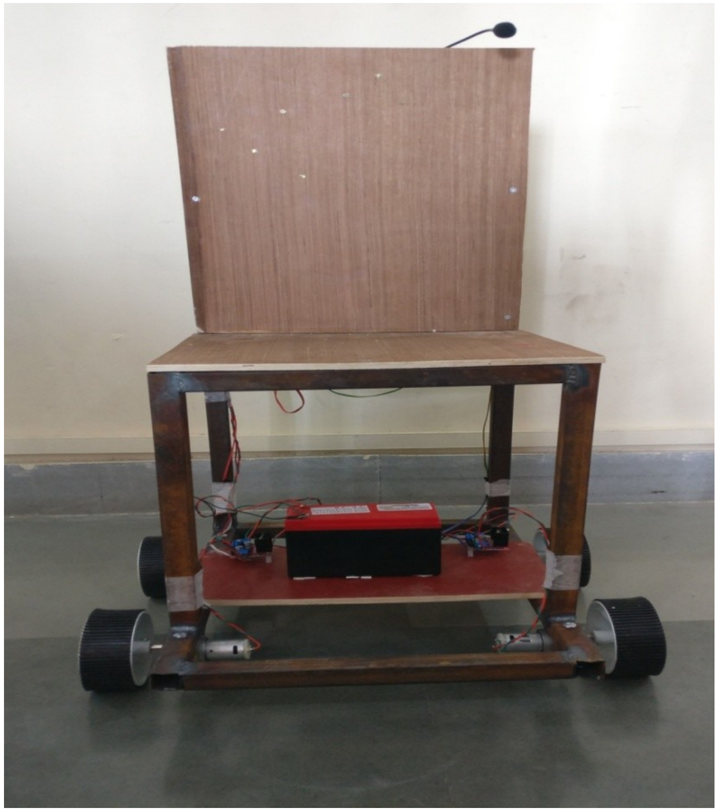


Fig 4.1.4.2.3 Actual Implementation

# 5. TESTING

## 5.1 INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design, and code generation. The increasing visibility of software as a system element and the attendant costs associated with a software as a system failure are motivating forces for well-planned, through testing. It is not unusual for a software development organization to expand between 30 and 40 percent of total project effort on testing. In the extreme, testing of human-related software (e.g., Flight Control, Nuclear Reactor monitoring) can cost three to five times as much as all other software engineering steps combined! Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that uncovers an error.

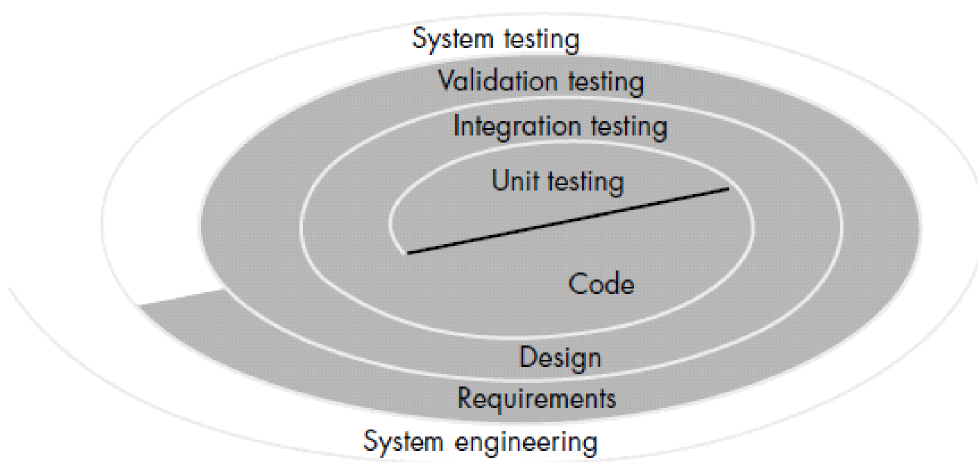



Fig 5.1.1 Software Testing

The software testing strategy can also be considered in the backdrop of the spiral. Unit Testing begins at the counter flow of the spiral and stresses on each and every component of the software as implemented using the source code. The testing progresses as shown in figure. It progresses right from the innermost shell i.e. unit testing moving outward along the spiral to integration testing, where the cornerstone is on design and the construction of the software architecture. Going further outward along the spiral comes validation testing and so on.

At the final stage of the spiral comes the system testing. In which the software and other system elements are tested together. If there is a need to test a computer software this spiral out along the streamlines that broadens the scope of testing.

## 5.2 VALIDATION TESTING

Validation testing is the process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the user's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

### 5.2.1 Validating Voice training :

Training of voice is simple. User just needs to set the VR3 in training mode. It accepts the voice in 2 stages. In first one, it accepts a command and in next one it revalidates it. When training, the two LEDs on the Voice Recognition Module can indicate the training process. After sending the training command, the SYS_LED (yellow) is blinking fast which remind you to get ready. Speak your voice command as soon as the STATUS_LED (red) light lights on. The recording process ends once when the STATUS_LED (red) lights off. Then the SYS_LED is blinking again, get ready for next recording process. When the training process ends successful, SYS_LED and STATUS_LED blink together. If the training fails, SYS_LED and STATUS_LED blink together, but quickly.

In the system, the commands have been checked multiple times so that the voice recognition module is trained perfectly. The module was trained in noise-free environment to maintain accuracy of the system. The trained commands were later checked and validated by giving commands in a little noise to check the degree of accuracy of the system and it was found that the system can accept noise to a low degree but might fail to operate in a desired manner in noise of higher degree.

```
Signature:t
--------------------------------------------------------------------------------
sigtrain 0 on
--------------------------------------------------------------------------------
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       No voice
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Cann't matched
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Cann't matched
Record: 0       Speak now
Record: 0       Speak again
Record: 0       Success
Success: 1
Record 0        Trained
SIG: on
--------------------------------------------------------------------------------
sigtrain 1 off
--------------------------------------------------------------------------------
Record: 1       Speak now
Record: 1       Speak again
Record: 1       Cann't matched
Record: 1       Speak now
Record: 1       Speak again
Record: 1       Success
Success: 1
Record 1        Trained
SIG: off
```

Fig 5.2.1.1 Voice train validation

### 5.2.2 Validating voice recognition

Once all the commands are loaded, the recognition of the stored command is started. Since the VR3 is a speaker dependent module, it only accepts the commands trained by the particular user. If the intonation of the user does not match with the stored command, the system doesn’t respond but if another person with the same intonation gives the command, the system moves according to the command given.

### 5.2.3 Validating connections

For training, the connections of Arduino and the voice recognition module is as follows:

| Arduino | | VR Module |
|---|---|---|
| 5V | ----> | 5V |
| 2 | ----> | TX |
| 3 | ----> | RX |
| GND | ----> | GND |

Fig 5.2.3.1 Connection between Arduino and Vr3.

If the connections are not according to above figure, the whole interfacing would be hampered.

## 5.3 Integration Testing

Integration testing is the phase in testing in which individual modules are combined and tested as a group. In our system, if the hardware and software are not interfaced together properly then the system does not respond. If the voice stored in the software does not match with the given voice, then the wheelchair would not move. There are 3 modules in the system namely Voice recognition module, Command display module and motor driving module. Each module is interfaced using Arduino Uno.

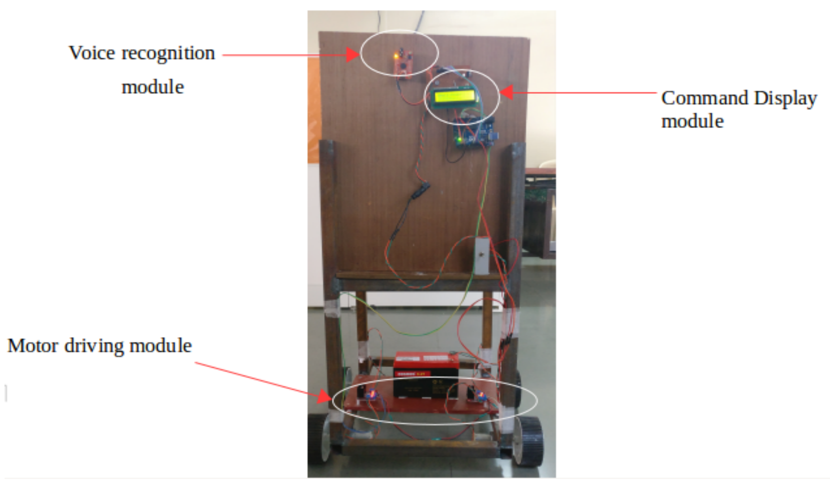



Fig 5.3.1 Integrated modules

### 5.4 Performance testing

Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

#### 5.4.1 Load testing

It is the simplest form of testing conducted to understand the behavior of the system under a specific load. For our system, the motors used in the system have stall torque approximately equal to 20-25kgs and since we have used 4 motors, the weight gets divided by four. Thus, the maximum weight that can be handled by the wheelchair is 30-35 kgs. If the input weight is more than this, then the wheelchair might not perform the desired wheelchair properly. Thus, the movement (speed) of wheelchair depends on the weight imposed on chair.

Among 5 commands(go, back, left, right and stop), go and back commands would give similar movements but different speeds but the commands left and right would give ambiguous results. The degree of angle the chair has rotated under left or right command would be different for different weights. For example, an empty chair would move approximately 300° whereas under a weight of 10 kgs the chair would rotate 130° hence, different angles of rotation according to imposed weights can be given as:

| Imposed weight(in kgs) | Angle of rotation(in degree) |
|---|---|
| 0 | 190 |
| 5 | 185 |
| 7 | 170 |
| 10 | 155 |
| 12 | 130 |
| 14 | 110 |
| 18 | 105 |
| 24 | 95 |
| 30 | 90 |
| 35 and above | 80 |

Table 5.4.1.1 Angle of rotation of motors according to imposed weight

# 6. CONCLUSION AND FUTURE SCOPE

The Voice Controlled Wheelchair comprises the interfacing of Arduino Uno R3 and the Voice recognition kit V3 thus making it easy for the physically disabled people to move about without anyone's help. The only thing needed to ride the wheelchair is the synthetic voice commands of the person through the unilateral microphone to give more flexibility to the users. The hardware used in the system are voice recognition kit, Arduino Uno r3, motor drivers, D.C. motors, unilateral microphone, LCD display and for the proper current supply a battery is also installed at the bottom.

The design and implementation of a voice controlled wheelchair for disabled people using voice recognition processor for controlling the motion of a wheelchair is shown. The direction of the wheelchair now can be selected using the specified voice commands. The design not only reduce the manufacture cost compared with present market but also will give great competitive with other types of electrical wheelchair.

The only thing needed to ride the wheelchair is the voice of the person. By implementing such a system we directly enhance the lifestyle of a physically disabled person in the community. This project has many advantages like safety, comfort, energy saving, full automation etc.

In future, the system can also be enhanced by using the hand gestures technology to recognize the movements.  Obstacle avoidance system can also be developed and added in the system for smoother functioning.

# 7. REFERENCES

[1]R.Puviarasi, Mritha Ramalingam, Elanchezhian Chinnavan, Low Cost Self-assistive Voice Controlled Technology for Disabled People, International Journal of Modern Engineering Research (IJMER), 4, Jul.-Aug. 2013.

[2]G Azam and M T Islam, Design, Fabrication of a Voice Controlled Wheelchair for Physically Disabled People, International Conference on Physics Sustainable Development & Technology (ICPSDT-2015), 19-20,August. 2015

[3] Ms. S. D. Suryawanshi Mr. J. S. Chitode Ms. S. S. Pethakar, Voice Operated Intelligent Wheelchair, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 5, May 2013

[4] Kharka Bahadur Rai, Jeetendra Thakur, Nirmal Rai, Voice Controlled Wheelchair using Arduino, International Journal of Science, Technology & Management,
Volume No.04, Issue No. 06, June 2015

[5] Ali A. Abed, Design of Voice Controlled Smart Wheelchair, International Journal of Computer Applications Volume 131 – No.1, December 2015.

[6] Linda Fehr, MS. W. Edwin Langbein, Steven B. Skaar., Journal of Rehabilitation Research and Development. Adequacy of power Wheelchair control interfaces for persons with severe disabilities: A clinical survey, 2000

[7] Simpson RC, Levine SP, Voice control of a powered wheelchair, IEEE Trans Neural System Rehabilitation Eng. 2000, 122-125.

[8] Aakash. A. Hongunti, Mayuri.Deulkar, Varsha Sable, Voice Operated Intelligent Wheelchair, International Journal of Advanced Research in Computer Science and Software Engineering,4, April 2014,

# ANNEXURE I: S/W PROGRAM DETAILS

The coding part involves interfacing of the Arduino and the motors i.e. the movement of motors in a particular direction as per the given command. The coding is done in C language.

```
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

#define motorDDR  DDRC
#define motorPORT PORTC
#define m1in1     PC0
#define m1in2     PC1
#define m2in1     PC2
#define m2in2     PC3

LiquidCrystal lcd(8, 9, 10, 11, 12, 13);
SoftwareSerial mySerial(2, 3);

char str[16];
char rbuff[16];
char ch;

void setup() {
  motorDDR |= (1 << m1in1 | 1 << m1in2 | 1 << m2in1 | 1 << m2in2);
  motorPORT &= ~(1 << m1in1 | 1 << m1in2 | 1 << m2in1 | 1 << m2in2);

  Serial.begin(115200);
  mySerial.begin(9600);
  // put your setup code here, to run once:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("Whell Chair");
  delay(2000);

  mySerial.write(0xAA);
  mySerial.write(0x07);
  mySerial.write(0x30);
  mySerial.write((byte)0x00);
  mySerial.write(0x01);
  mySerial.write(0x02);
  mySerial.write(0x03);
  mySerial.write(0x04);
  mySerial.write(0x0A);
  /*
  mySerial.write(0xAA);
  mySerial.write(0x07);
  mySerial.write(0x30);
```

```
  mySerial.write(0x07);
  mySerial.write(0x08);
  mySerial.write(0x09);
  mySerial.write(0x0A);
  mySerial.write(0x0B);
  mySerial.write(0x0A);
  */
}

void loop() {
  // put your main code here, to run repeatedly:
  /*if(mySerial.available()) {
    ch = mySerial.read();
    switch(ch) {
      case 'f':
                forward();
                break;
      case 'b':
                reverse();
                break;
      case 'l':
                left();
                break;
      case 'r':
                right();
                break;
      case 's':
                wheel_stop();
                break;
    }
    mySerial.write(ch);
  }*/
  if (mySerial.available()) {
    delay(100);
    int r = mySerial.available();

    sprintf(str,"r:%d",r);
    //lcd.setCursor(0, 0);
    //lcd.print(str);

    mySerial.readBytes(rbuff, r);
    lcd.setCursor(0, 1);
    //lcd.print(rbuff+7);

    if(strstr(rbuff+7,"go") != 0) {
      lcd.print("forward");
      forward();
      delay(5000);
      wheel_stop();
    }
    else if(strstr(rbuff+7,"back") != 0) {
```

```
      lcd.print("reverse");
      reverse();
      delay(5000);
      wheel_stop();
    }
    else if(strstr(rbuff+7,"left") != 0) {
      lcd.print("left   ");
      left();
      delay(2000);
      wheel_stop();
    }
    else if(strstr(rbuff+7,"right") != 0) {
       lcd.print("right  ");
       right();
       delay(2000);
       wheel_stop();
    }
    else if(strstr(rbuff+7,"stop") != 0) {
       lcd.print("stop   ");
       wheel_stop();
    }
  }
}

void forward(void) {
  motorPORT |= (1 << m1in1 | 1 << m2in1);
  motorPORT &= ~( 1 << m1in2 | 1 << m2in2);
}

void reverse(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m2in1);
  motorPORT |= ( 1 << m1in2 | 1 << m2in2);
}

void left(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m2in2);
  motorPORT |= ( 1 << m1in2 | 1 << m2in1);
}

void right(void) {
  motorPORT |= (1 << m1in1 | 1 << m2in2);
  motorPORT &= ~( 1 << m1in2 | 1 << m2in1);
}

void wheel_stop(void) {
  motorPORT &= ~(1 << m1in1 | 1 << m1in2 | 1 << m2in1 | 1 << m2in2);
}
```

# ANNEXURE II

## PROJECT MEMBERS DETAILS

| SR.NO | NAME | CONTACT NUMBER | EMAIL ID |
|---|---|---|---|
| 1. | Navya Vijay | +91-77580337711 | navyavijay1995@gmail.com |
| 2. | K. Bhagyashree Rao | +91-9561471259 | kbhagyashree.rao@gmail.com |
| 3. | Riddhi Navsalkar | +91-9763728137 | rnavsalkar@gmail.com |
| 4. | Payal Bisen | +91-9405276683 | b.payal10@gmail.com |
| 5. | Tanushri Bajganiya | +91-7276249434 | tanubajgania11@gmail.com |