ABSTRACT

The main objective of SMART WHEELCHAIR project is to provide a fully functional project helpful for all our physically disabled people. The system is designed to control a wheel chair using manual keypad, voice, mobile gyro, joystick module. The objective of this project is to facilitate the movement of people who are disable or handicapped and elderly people who are not able to move well. The goal of this system will allow certain people to live a life with less dependence on others for their movement as a daily need. This can be realized and optimized with use the smart phone device as an intermediary or interface. In this project interfaces have been designed therefore to develop a program for controlling the movement of chair and an application which can handle or manage the graphical commands. This project uses Arduino UNO Microcontroller circuit and 4 TT DC motors to create the movement of wheel chair and Ultrasonic Sensors to detect the hurdles in between wheelchair and the way of direction. The results and analysis of this innovation will describe in this report. The result shows the module can be used for the future research works and to design excellence innovation that meets market needs and public interest.

KEYWORDS: Android Application, Wheel chair, physically Challenged, Ultra sonic Sensor, Voice Command, HC-05 Bluetooth Module, DC Motors, Arduino UNO Micro-controller

CHAPTER 1:- INTRODUCTION

INTRODUCTION

1.1 Basic Introduction

In this project we are using Android Application and joystick module for movement in wheelchair. But many of individuals with disabilities who need wheelchairs are satisfied with it, few members of the disabled community find it is difficult or impossible for operating a standard power wheelchair. This project is included in assistive technology. For handicapped and depended disable it is more independent, productive and enjoyable living.

To perform functions a handicapped person with locomotive disabilities needs a wheelchair that require him or her to move around. He/She can do so manually by pushing the wheelchair with his/her hands. However many of us have weak upper limbs or find the manual mode of operating too tiring. Therefore it is desirable to provide them with a motorized wheelchair which is controlled by moving a voice commands. Since motorized wheelchair is important that it be able to avoid obstacles automatically in real time, it can move at a fair speed. Cost of this motorized wheelchair is affordable for many handicapped people as possible, as well as for organizations that support it. With these requirements in mind we propose an automated wheelchair with real-time Herald avoidance capability. The power wheelchair control interfaces currently still not enough to provide mobility for substantial number of person with disabilities. Through research and design wise, the wheelchair to control development along safe and effective use of the provision independence and selfuse mobility. This project will provide disability weight innovative solutions to handle the wheel chairs to use mobile device.

This project describes a wheelchair which can be controlled only by using the android application and user's voice also. The main aim of this project is to facilitate the movement of the disabled people and elderly people who cannot

move properly so with this we can enable them to lead better lives without any problem. Mobile application is a key technology which can provide human interaction with machines for controlling a wheelchair. This project includes two parts which is software and hardware. It is realized that for input of our mobile we are using Android phone as an intermediary. In this project, Arduino UNO (Atmega 328) is used as controller to control the movement of wheelchair based on the human voice as an input. There are five basic movements of a wheelchair to be applied by the user. The Five operations perform by the wheelchair are described as following:

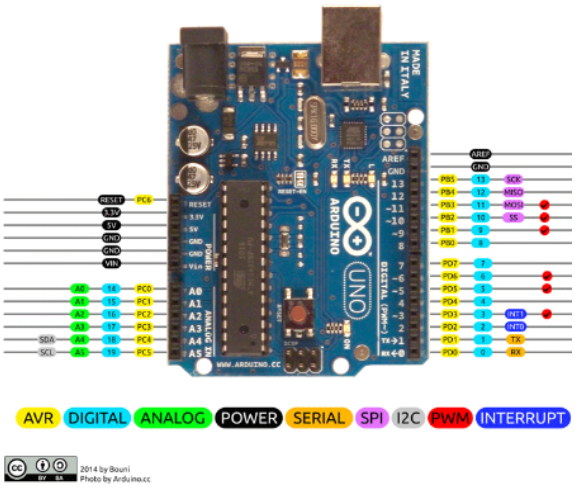
1) forward
2) backward
3) right
4) left
5) Stop condition

1.2 All Components Details

1.2.1 Hardware Components Used

1.2.1 Arduino UNO

Arduino Uno



Arduino Uno Pinout

Arduino Uno is a popular microcontroller development board based on 8-bit ATmega328P microcontroller. Along with ATmega328P MCU IC, it consists other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller.

Pin Category	Pin Name	Details
Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/Output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

Overview

Arduino Uno is a microcontroller board based on 8-bit ATmega328P microcontroller. Along with ATmega328P, it consists other components such as crystal oscillator, serial communication, voltage regulator, etc. to support the microcontroller. Arduino Uno has 14 digital input/output pins (out of which 6 can be used as PWM outputs), 6 analog input pins, a USB connection, A Power barrel jack, an ICSP header and a reset button.

How to use Arduino Board

The 14 digital input/output pins can be used as input or output pins by using pinMode(), digitalRead() and digitalWrite() functions in arduino programming. Each pin operate at 5V and can provide or receive a maximum of 40mA current, and has an internal pull-up resistor of 20-50 KOhms which are disconnected by default. Out of these 14 pins, some pins have specific functions as listed below:

- **Serial Pins 0 (Rx) and 1 (Tx):** Rx and Tx pins are used to receive and transmit TTL serial data. They are connected with the corresponding ATmega328P USB to TTL serial chip.
- **External Interrupt Pins 2 and 3:** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- **PWM Pins 3, 5, 6, 9 and 11:** These pins provide an 8-bit PWM output by using analogWrite() function.
- **SPI Pins 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK):** These pins are used for SPI communication.
- **In-built LED Pin 13:** This pin is connected with an built-in LED, when pin 13 is HIGH – LED is on and when pin 13 is LOW, its off.

Along with 14 Digital pins, there are 6 analog input pins, each of which provide 10 bits of resolution, i.e. 1024 different values. They measure from 0 to 5 volts but this limit can be increased by using AREF pin with analog Reference() function.

- Analog pin 4 (SDA) and pin 5 (SCA) also used for TWI communication using Wire library.

Arduino Uno has a couple of other pins as explained below:

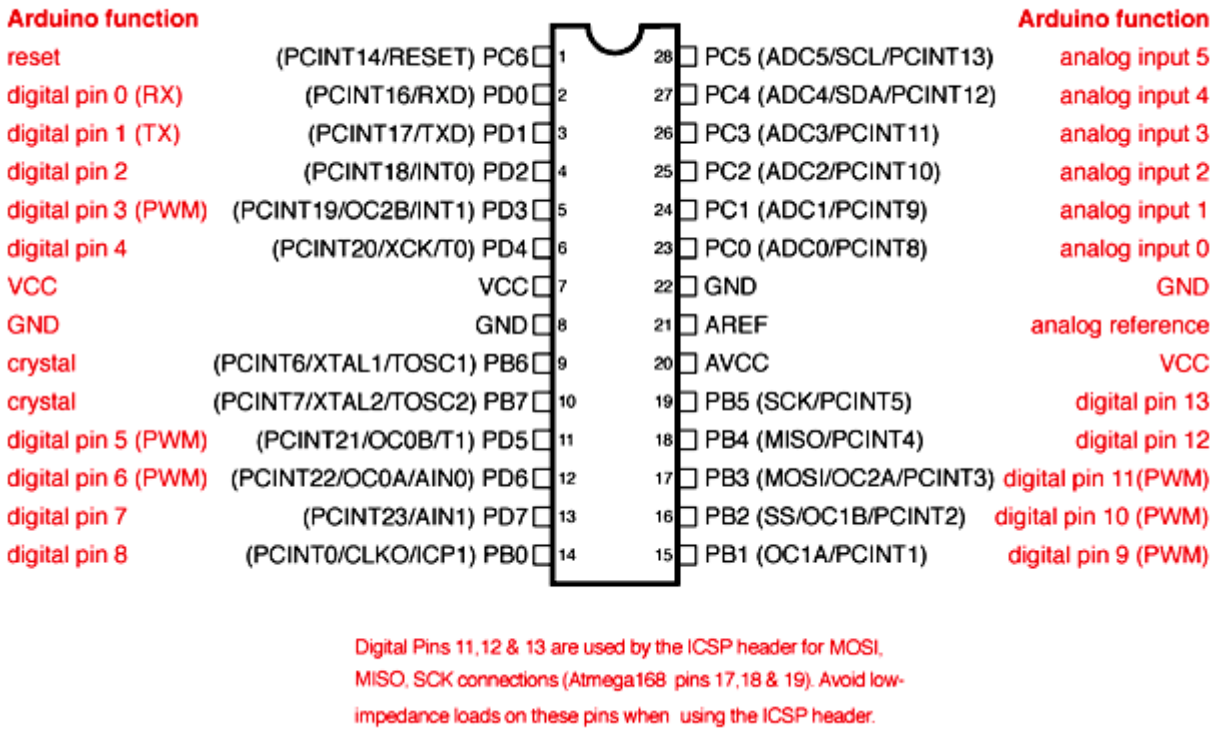
- **AREF:** Used to provide reference voltage for analog inputs with analogReference() function.
- **Reset Pin:** Making this pin LOW, resets the microcontroller.

Communication

Arduino can be used to communicate with a computer, another Arduino board or other microcontrollers. The ATmega328P microcontroller provides UART TTL (5V) serial communication which can be done using digital pin 0 (Rx) and digital pin 1 (Tx). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The ATmega16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. There are two RX and TX LEDs on the arduino board which will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328P also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

Arduino Uno to ATmega328 Pin Mapping

When ATmega328 chip is used in place of Arduino Uno, or vice versa, the image below shows the pin mapping between the two.



Software

Arduino IDE (Integrated Development Environment) is required to program the Arduino Uno board.

1.2.2 HC- 05 Bluetooth module

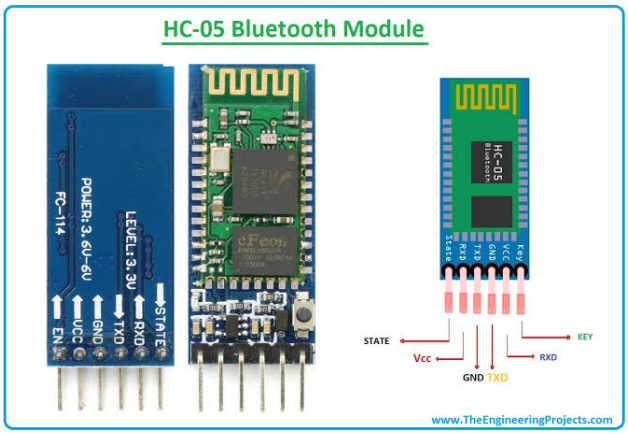


Figure 2 :- HC-05 Bluetooth Module

HC-05 module is an easy to use Bluetooth SPP(Serial Port Protocol) module, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication. This serial port bluetooth module is fully qualified Bluetooth V2.0+EDR(Enhanced Data Rate)3Mbps Modulation with complete 2.4 GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping

Feature).

The Bluetooth module HC-05 is a MASTER/SLAVE module.By default the factory setting is SLAVE. The Role of the module(Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

4.3.1 Pin Description

The HC-05 Bluetooth Module has 6pins. They are as follows:

● Enable: When enable is pulled LOW, the module is disabled which means the module will not turn on and it fails to communicate.When enable is left open or connected to 3.3V, the module is enabled i.e the module remains on and communication also takes place.

● Vcc: Supply Voltage 3.3V to 5V

● GND: Ground pin

● TXD & RXD: These two pins acts as an UART interface for communication

● State: It acts as a status indicator.When the module is not connected to paired with any other bluetooth device,signal goes Low.At this low state,the led flashes continuously which denotes that the module is not paired with other device.When this module is connected to/paired with any other bluetooth device,the signal goes High.At this high state,the led blinks with a constant delay say for example 2s delay which indicates that the module is paired.

● Button Switch: This is used to switch the module into AT command mode.To enable AT command mode,press the button switch for a second.With the help of AT commands,the user can change the parameters of this module but only when the module is not paired with any other BT device.If the module is connected to any other bluetooth device, it starts to communicate with that device and fails to work in AT command mode.

4.3.2 HC-05 Default Settings

● Default Bluetooth Name: HC-05

- Default Password: 1234 or 0000
- Default Communication: Slave
- Default Mode: Data Mode
- Data Mode Baud Rate: 9600, 8, N, 1
- Command Mode Baud Rate: 38400, 8, N, 1
- Default firmware: LINVOR

1.2.3 IR Sensor

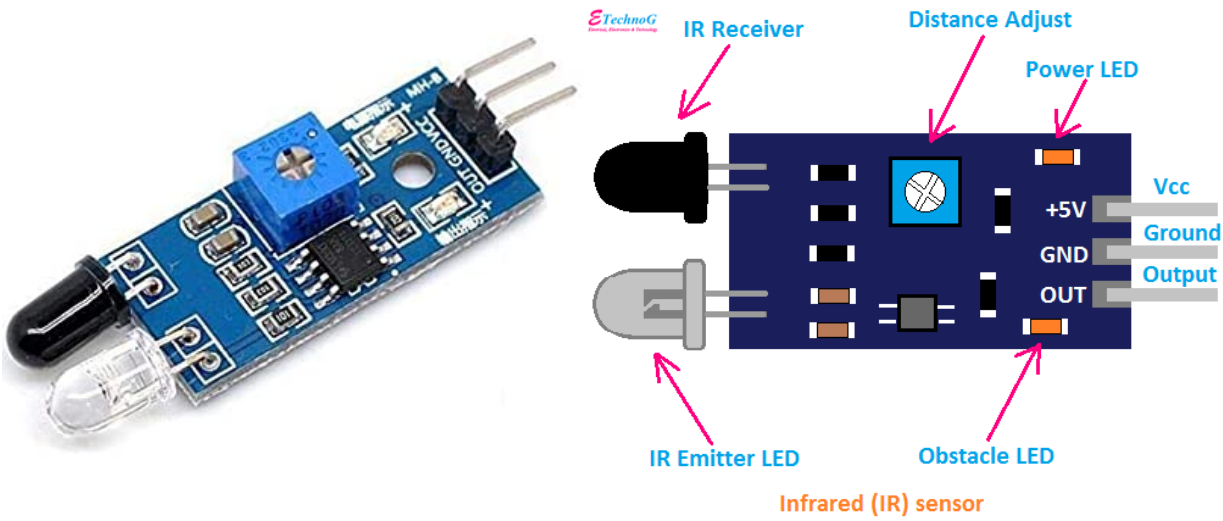


Figure 3 :- IR Sensor Module

The IR sensor module consists mainly of the IR Transmitter and Receiver, Op-amp, Variable Resistor (Trimmer pot), output LED along with few resistors.

IR LED Transmitter

IR LED emits light, in the range of Infrared frequency. IR light is invisible to us as its wavelength (700nm – 1mm) is much higher than the visible light range. IR LEDs have light emitting angle of approx. 20-60 degree and range of approx. few centimeters to several feets, it depends upon the type of IR transmitter and the manufacturer. Some transmitters have the range in kilometers. IR LED white or transparent in colour, so it can give out amount of maximum light.

Photodiode Receiver

Photodiode acts as the IR receiver as its conducts when light falls on it. Photodiode is a semiconductor which has a P-N junction, operated in Reverse Bias, means it start conducting the current in reverse direction when Light falls on it, and the amount of current flow is proportional to the amount of Light. This property makes

it useful for IR detection. Photodiode looks like a LED, with a black colour coating on its outer side, Black colour absorbs the highest amount of light.

LM358 Opamp

LM358 is an Operational Amplifier (Op-Amp) is used as voltage comparator in the IR sensor. the comparator will compare the threshold voltage set using the preset (pin2) and the photodiode's series resistor voltage (pin3).

Photodiode's series resistor voltage drop > Threshold voltage = Opamp output is High

Photodiode's series resistor voltage drop < Threshold voltage = Opamp output is Low

When Opamp's output is **high** the LED at the Opamp output terminal **turns ON** (Indicating the detection of Object).

Variable Resistor

The variable resistor used here is a preset. It is used to calibrate the distance range at which object should be detected.

IR Sensor Module Features

- 5VDC Operating voltage
- I/O pins are 5V and 3.3V compliant
- Range: Up to 20cm
- Adjustable Sensing range
- Built-in Ambient Light Sensor

- 20mA supply current
- Mounting hole

IR Sensor Module Pinout Configuration

Pin Name	Description
VCC	Power Supply Input
GND	Power Supply Ground
OUT	Active High Output

1.2.4 BO motors

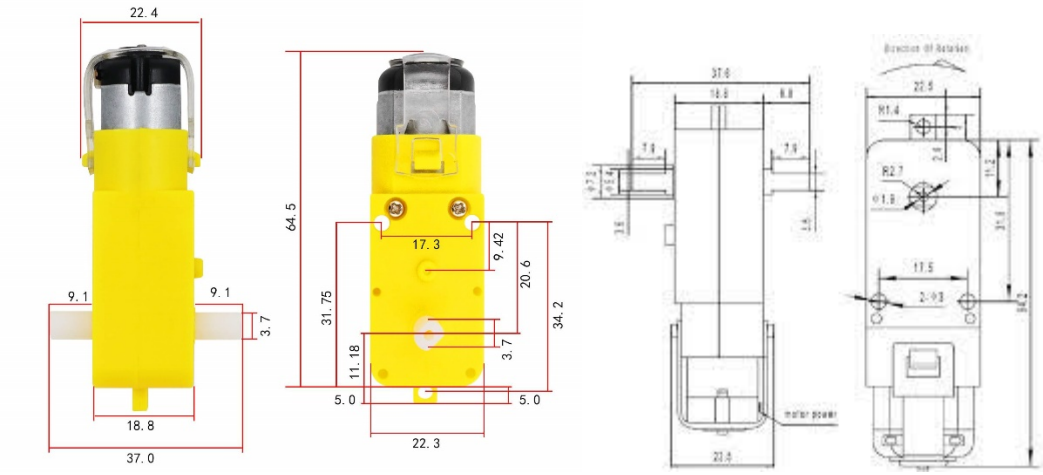


Figure 7 :- 150Rpm Bo Motor

The BO Series 2 150RPM DC Motor Plastic Gear Motor – BO series straight motor gives good torque and rpm at lower operating voltages, which is the biggest advantage of these motors.

Small shaft with matching wheels gives an optimized design for your application or **robot**. Mounting holes on the body & light weight makes it suitable for in-circuit placement. This **motor** can be used with 69mm Diameter **Wheel** for Plastic Gear Motors and 87mm Diameter Multipurpose **Wheel** for Plastic Gear Motors. Low-cost geared DC Motor. It is an alternative to our metal gear DC motors. It comes with an operating voltage of 3-12V and is perfect for building small and medium robots. Available with 60 and 150 RPM.

The motor is ideal for DIY enthusiasts. This motor set is inexpensive, small, easy to install, and ideally suited for use in a mobile robot car. They are commonly used in our 2WD platforms.

The 150 RPM Single Shaft L- Shaped BO Motor motor gives good torque and rpm at lower operating voltages, which is the biggest advantage of these motors.

Small shaft with matching wheels gives an optimized design for your application or robot. Mounting holes on the body & light weight makes it suitable for in-circuit placement. This motor can be used with 69mm Diameter Wheel for Plastic Gear Motors.

It is an alternative to our metal gear DC motors. It comes with an operating voltage of 3-12V and is perfect for building small and medium robots.

The motor is ideal for DIY enthusiasts. This motor set is inexpensive, small, easy to install, and ideally suited for use in a mobile robot car. They are commonly used in our 2WD platforms.

Specifications of 150 RPM Single Shaft L- Shaped BO Motor:-

- Shaft length: 7 mm
- Motor Design: L-Shaped
- Shaft Diameter: 5.5 mm
- Size: 55 x 48 x 23 mm.
- Operating Voltage: 3 to 12V
- Current (without loading): 40-180mA
- RPM: 150 rpm
- Output Torque: 0.8 kg cm.

Features of 150 RPM Single Shaft L- Shaped BO Motor:-

- Cost-effectiveness of the injection-molding process.
- Elimination of machining operations.
- Low density: lightweight, low inertia.

- Uniformity of parts.
- Capability to absorb shock and vibration as a result of elastic compliance.
- Ability to operate with minimum or no lubrication, due to inherent lubricity.
- The relatively low coefficient of friction.
- Corrosion-resistance; elimination of plating, or protective coatings.
- The quietness of operation.
- Tolerances are often less critical than for metal gears, due in part to their greater resilience.
- Consistency with the trend to greater use of plastic housings and other components.

Operating Voltage (VDC)	3 ~ 12
Shaft Length (mm)	8.5
Shaft Diameter (mm)	(Double D-type) 5.5
No Load Current (mA)	40-180mA.
Rated Speed After Reduction (RPM)	150
Rated Torque (Kg-Cm)	0.8
Weight (gm)	30
Dimensions in mm (LxWxH)	42 × 55 × 22
Gearbox Shape	L-Shape
Shipment Weight	0.033 kg
Shipment Dimensions	6 × 6 × 2 cm

Figure 8 :- Bo motors Specs

1.2.6 18650mAp Battery and Charger

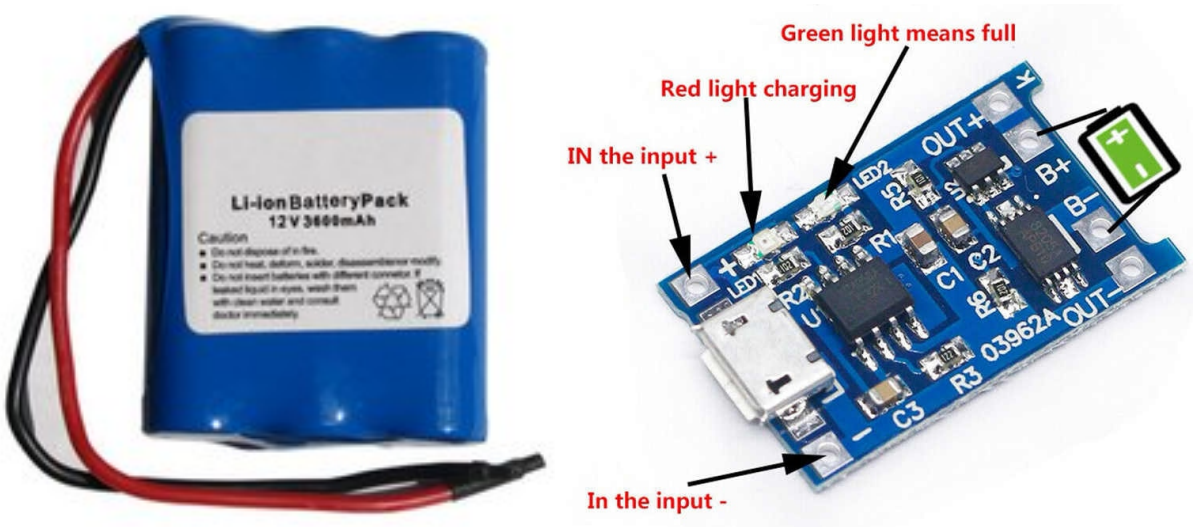


Figure 9 :- Li-ion Battery and Charger

The 18650 battery is a Li-ion battery named after its 18mm × 65mm cylindrical size (diameter × height). When compared to AA size, it's height and diameter both are larger. They are not replacements for AA or AAA size cells.
The battery type 18650 are popular in rechargeable and high current draining devices considering its high-level capabilities like 250+ charge cycle and higher energy density.

The 18650 Li-ion battery due to its adaptability all around can be found in various fields, say, electric cars/ scooters, power banks, utility gadgets such as emergency lamps, torchlight, etc.
Its safety property along with its high output current and energy capacity makes this battery popular among tech industries.

The 18650 battery specification includes its properties like the voltage, capacity, charge-discharge cycle, output current, output voltage and so on.

This is a generalized specification of 18650 Li-ion battery, only properties marked with the remark of “Standard” are common to all 18650 batteries else not.

This is to note that other properties such as capacity charge discharge cycles and capacity (mAh) are subjected to independent technologies to increase capability values to the consumers by doing research and development. All the 18650 specs are listed in table with detail. You might like this article on power bank.

18650 battery size / dimension:
The Standard 18650 battery size is 18 ✖ 65mm.
The 18650 battery length is 65mm.
The diameter of the 18650 battery is 18mm.
To be more precise, it has an approximate length of 65mm and an approximate diameter is 18mm but technically 18650 battery size is allowed with some tolerance in length and diameter.
Thus you could find specification written as, (say) 18 ± 0.41mm ✖ 65 ± 0.25mm on datasheet and features of li-ion cell.

Just remember 18 ✖ 65mm as a standard size, remaining things will be taken care of by designers and manufacturers of devices and batteries.
The size of the 18650 battery is allowed for the reason that, various gadgets or devices inhibit different locally developed technologies. That might not able to manufacture exact precise length & diameter of either batteries or battery holding space, to fit the device or 18650 Lithium battery respectively.

Properties	Specifications	Remark
Component type	Battery/ Cell	Power supply
Battery type	Li-ion	Technology
Model No	18650	-
Size/ dimension	18 ✕ 65 mm	Standard (dd ✕ hh)
Voltage	3.7 volt	Nominal (std.)
Capacity	1200-3600mah	Per cell
Operating voltage	2.5-4.2 volt	Range (std)
Cutoff voltage	2 - 2.5 volts	Check respective datasheet
Weight	30gms to 55gms	Depends on capacity & technology
Charge density (Energy per cell)	1.5 - 11.5Wh	Energy
Charge discharge cycle	250 to 2000	In number ()
Optimum /Minimum charging time	2.5 hrs to 3.5 hrs	Per cell
Shelf life	36+ months	-

Shelf life	36+ months	-
Charging method	CC and CV	-
Charging voltage	4.2v - 5v	5 volt max
Output current	-	Check C rating*
Charging current	-	Check C rating*
Internal resistance	-	High
Product variant	Regular and Flat top	-
Self discharge	6 - 10 %	Per month

Figure 10 :- Battery charger Specs

1.3.7 L293D motor driver Module

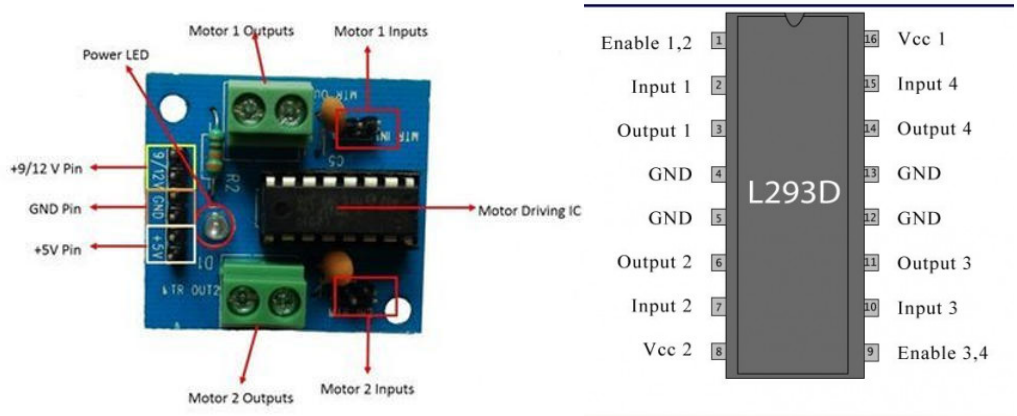


Figure 11 :- L293D motor Driver Module and IC

The L293 and L293D are quadruple high-current half-H drivers. These devices are designed to drive a wide array of inductive loads such as relays, solenoids, DC and bipolar stepping motors, as well as other high-current and high-voltage loads. All inputs are TTL compatible and tolerant up to 7 V. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications. On the L293, external high-speed output clamp diodes should be used for inductive transient suppression. On the L293D, these diodes are integrated to reduce system complexity and overall system size. A VCC1 terminal, separate from VCC2, is provided for the logic inputs to minimize device power dissipation. The L293 and L293D are characterized for operation from 0°C to 70°C.

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC. Dual H-bridge Motor Driver integrated circuit (IC).

Detailed Description:

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, Hence H-bridge IC are ideal for driving a DC motor. In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller. There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch.

Instructions/Manuals/Technical details:

VCC is the voltage that it needs for its own internal operation 5v; L293D will not use this voltage for driving the motor. For driving the motors it has a separate provision to provide motor supply VSS (V supply). L293d will use this to drive the motor. It means if you want to operate a motor at 9V then you need to provide a Supply of 9V across VSS Motor supply. The maximum voltage for VSS motor supply is 36V. It can supply a max current of 600mA per channel. Since it can drive motors Up to 36v hence you can drive pretty big motors with this l293d.

VCC pin 16 is the voltage for its own internal Operation. The maximum voltage ranges from 5v and upto 36v

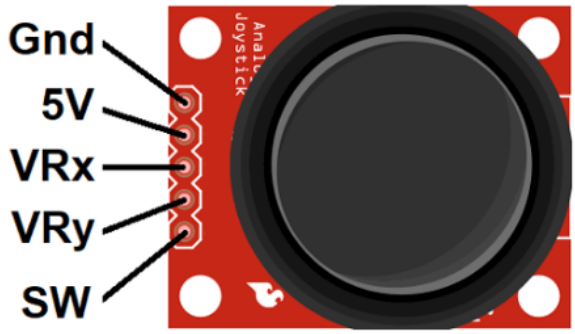
absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC1} (see Note 1)		36 V
Output supply voltage, V_{CC2}		36 V
Input voltage, V_I		7 V
Output voltage range, V_O		−3 V to V_{CC2} + 3 V
Peak output current, I_O (nonrepetitive, t ≤ 5 ms): L293		±2 A
Peak output current, I_O (nonrepetitive, t ≤ 100 µs): L293D		±1.2 A
Continuous output current, I_O: L293		±1 A
Continuous output current, I_O: L293D		±600 mA
Package thermal impedance, θ_{JA} (see Notes 2 and 3):	DWP package	TBD°C/W
	N package	67°C/W
	NE package	TBD°C/W
Maximum junction temperature, T_J		150°C
Storage temperature range, T_{stg}		−65°C to 150°C

Figure 12 :- L293D motor Driver Specs

1.3.9 Joystick Module

Joystick Module



Joystick Pinout

Pin Configuration

Pin No.	Pin Name	Description
1	Gnd	Ground terminal of Module
2	+5v	Positive supply terminal of Module
3	VRx	Voltage Proportional to X axis
4	VRy	Voltage Proportional to Y axis
5	SW	Switch

Features

- Two independent Potentiometer: one for each axis (X and Y)
- Auto return to center position
- Low weight
- Cup-type Knob

- Compatible to interface with Arduino or with most microcontrollers

Technical Specifications

- Operating Voltage: 5V
- Internal Potentiometer value: 10k
- 2.54mm pin interface leads
- Dimensions: 1.57 in x 1.02 in x 1.26 in (4.0 cm x 2.6 cm x 3.2 cm)
- Operating temperature: 0 to 70 °C

Internal Structure

The below image is the **internal diagram of a Joystick Module**. It consists of two Potentiometer, each for one axis (X and Y). Both 10k potentiometer are independent to move in their particular direction. SW (Switch) pin is connected to a push button internally.

Where Joysticks Are Used?

When we listen the word "**Joystick**" we think of Game controllers. If we talk about Electronics there are many useful application of Joystick. These type of module are mostly used in Arduino based DIY projects and Robot Control. As we know, the module gives analog output so it can be used for feeding the analog input based on direction or movement. It can also be connected to a movable camera to control its movement.

How to Use Joystick?

We can use a Joystick Module with Arduino, Raspberry Pi and any other Micro-controllers. We just have to connect the axis Pins VRx and VRy to the ADC Pins of the micro-controller. If you want to use the switch then connect it to the digital Pin of the Micro-controller. Follow the below block diagram to connect **Joystick Module with Microcontroller**.

CHAPTER 2:- REVIEW OF LITERATURE

Chapter 2

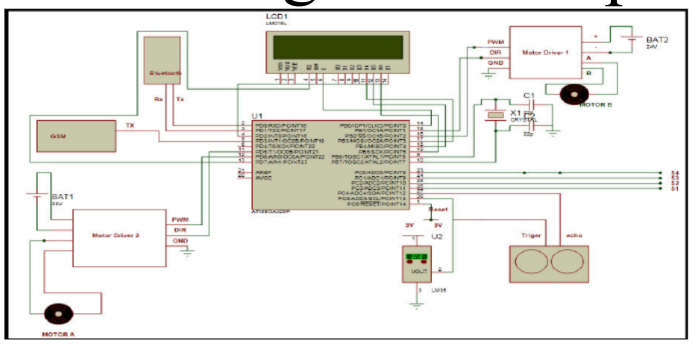
Review of Literature

2.1 Design of Voice Controlled Smart Wheelchair for Physically Challenged Persons

Design of Voice Controlled Smart Wheelchair for Physically Challenged Persons

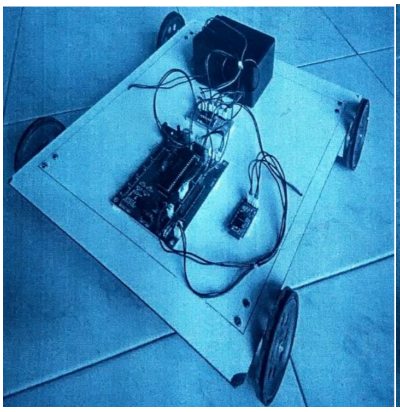
Physically challenged persons those who are suffering through different physical disabilities face many challenging problems in their day to day life for commutating from one place to another and even sometimes they need to have to be dependent on other person to move from one place to another. There have been many significant efforts over the past few years to develop smart wheel chair platforms that could enable the person for its ease of operation without any ambiguity. The main aim of our paper is to develop the smart wheel chair to make the life easier of physically challenged persons. This voice controlled smart wheel chair comes with enhanced features like electric powered, voice control, line follower with obstacle avoidance etc. The smart wheel chair control unit consists of integration of AVR microcontroller ATmega328 with Bluetooth module, GSM module SIM900, ultrasonic and infrared sensors, temperature sensor

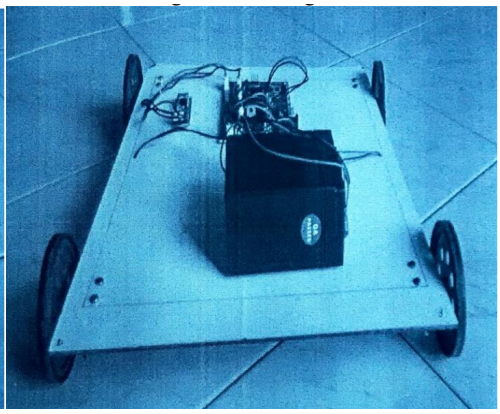
LM35 and motor driving circuit for controlling motor's speed.



2.2 VOICE CONTROLLED WHEEL CHAIR

Voice based system has been developed to control a mobile robot using oral commands.The system is divided in to two components as speech recognition engine, hidden Markov Models are used as the main method recognizing oral commands. The robotic control unit is developed under the ARIA environment, which is provided by the robot manufacturer. This paper describes about the Voice control wheel chair system by using speech recognition through Bluetooth module. The system is designed to control a wheelchair using the voice of consumer. The objective of this paper is to facilitate the movement of people who are disabled or handicapped and elderly people who are not able to move well. The result of this design will allow certain people to live a life with less dependence on others. Speech recognition technology is a key which may provide new way of human interaction with machines or tools. Thus the problem that they faced can be solved by using speech recognition technology to move the wheel chair. This can be realized with the Bluetooth as an intermediary. An interface is designed therefore to develop the program for recognizes s voice in turns control the movement of wheelchairs. The results and analysis of this innovation will describe in this report. The result shows the module can be used for the future research works and to design excellence innovation that meets market needs and public interest.






AMR_Voice
Android Meets Robots !
Talk to Your Robot
Click belowto Speak!
stop

CHAPTER 3:- THEORY, METHODOLOGY AND ALGORITHM

Chapter 3
Theory, Methodology and Algorithm

3.1. Hardware circuit diagram.

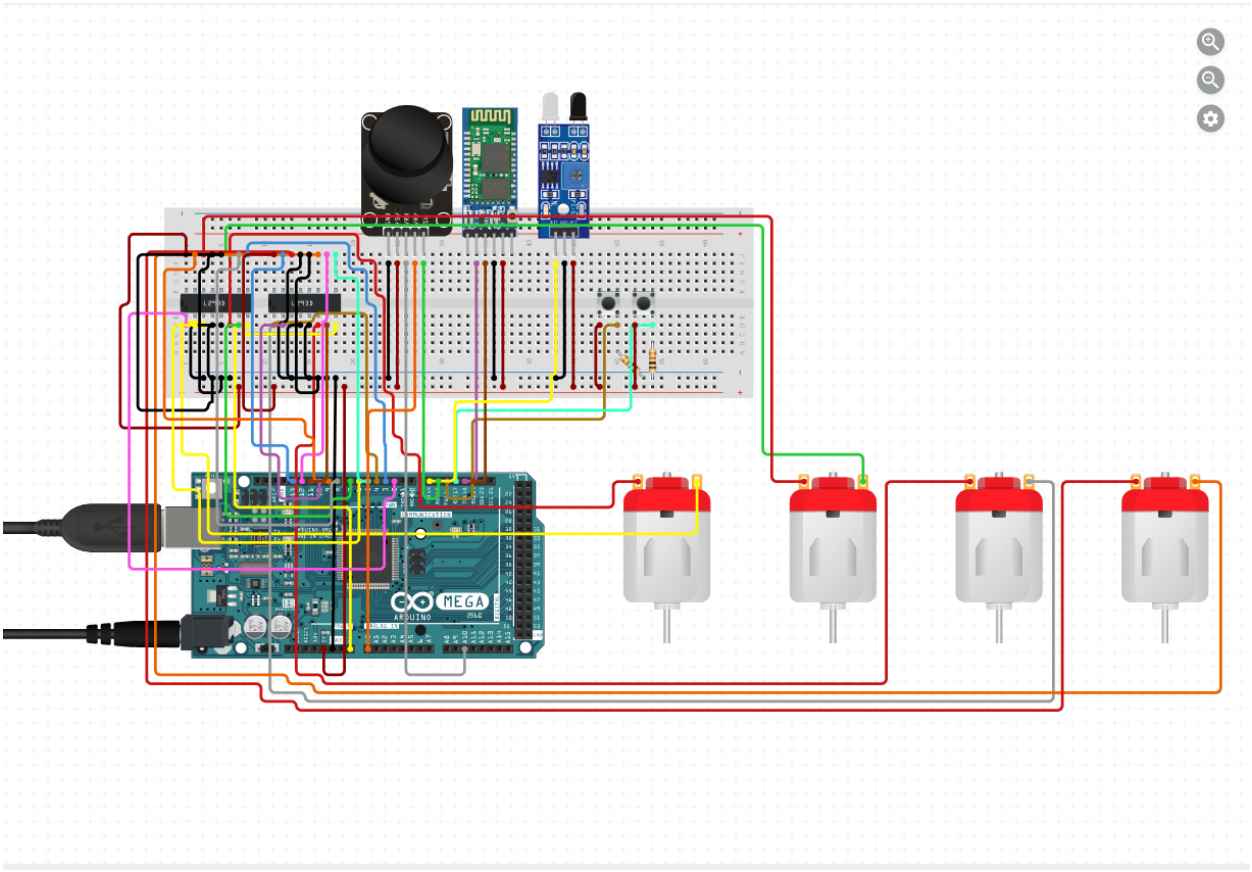


Figure 16 :- Project working circuit diagram

3.2. Hardware block diagram

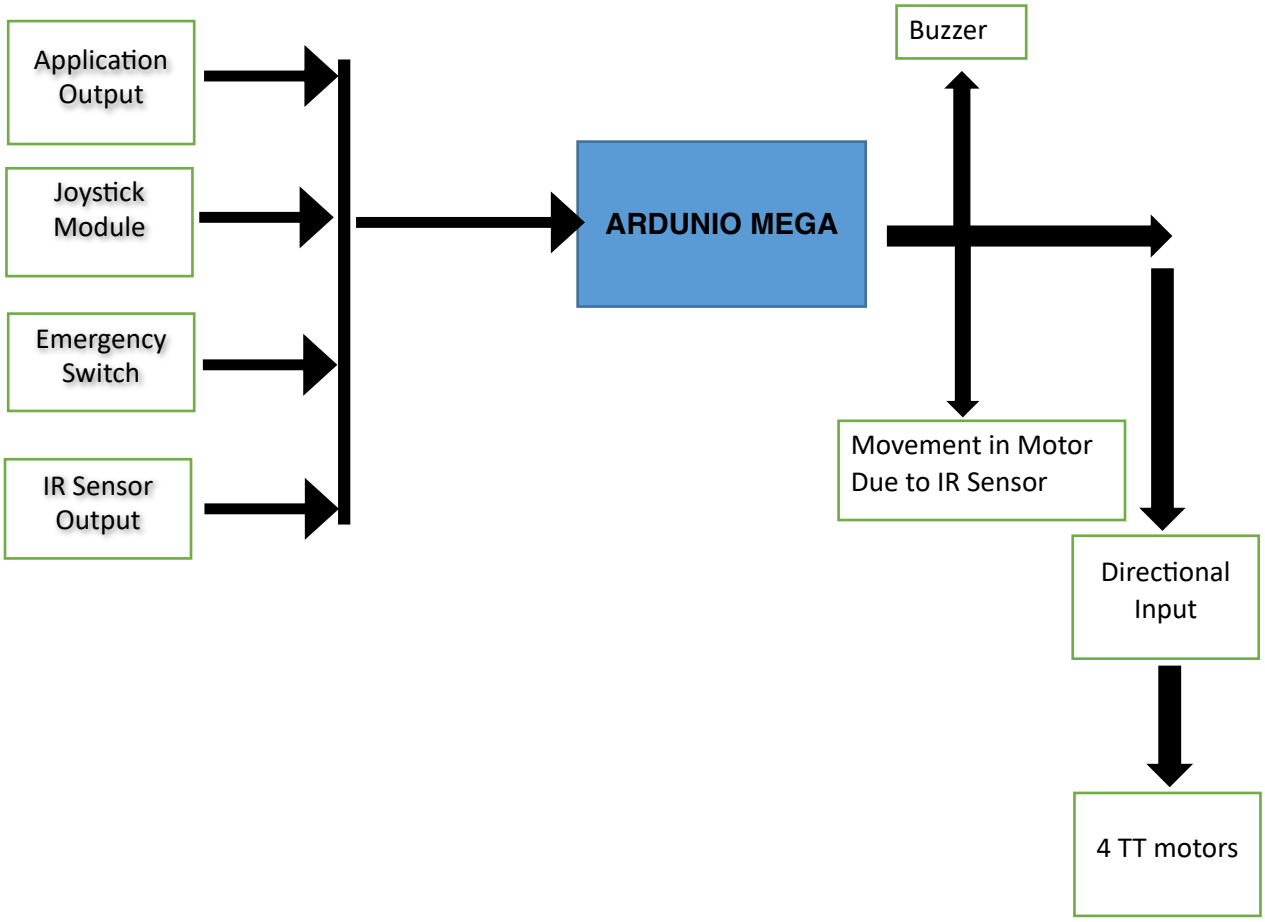


Figure 17 :- Project Block diagram

3.3 Hardware working details

It has a combination of 2 back wheel with BO motors interfaced which operates according to the input received from the arduino UNO. The data which is transferred to Arduino is from mobile application, which has a functionality to transfer data via Keypad, Voice and Mobile Gyro. The movement can be any command e.g Forward, Backward, Right Turn, Left Turn etc. Which will be further explained. It has function like location tracking for family members to track the position of the patient and a emergency Stop Button to avoid any accidents.

Working

Case 1 :- When we need to transfer data using Keypad on application:- For transferring data from mobile application to arduino using Keypad there are options available in keypad to switch in all the directions. The front key on keypad when pressed moves the wheelchair in forward direction and hence all the movement in wheel chair is obtained so on.

Case 2 :- When we need to transfer data using voice activation:- For transferring data from mobile application to arduino using Voice there are options available in mobile to activate it after activation when we give a command of "Forward" the wheelchair will start to move in forward direction and hence rest commands like left, right and backward will be performed. We have tried to implement hindi and Marathi commands like "Aage" for front, "Pechey" for backward and similar for rest of the commands. Same way we have made efforts for implementing Marathi commands too.

Case 3 :- When we need to transfer data using Mobile Gyro sensor :- For transferring data using Gyro sensor we need to activate it using mobile application where in we get the option to tilt our mobile device in either of 1 direction that can be right, Left and so on, so according to the tilt in gyro the wheel chair will move and will let us achieve the movement we are in need of.

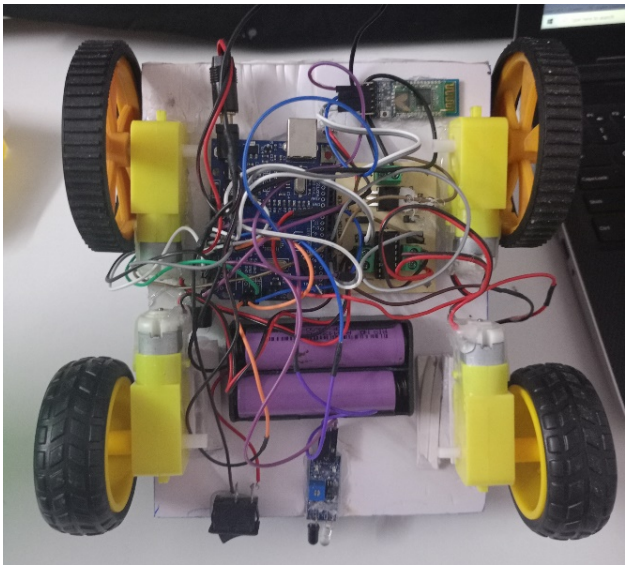
Case 4 :- Option for Tracking the wheelchair by family members from a long distance if required :- This is one of the advance feature of our project where in family members of the patient can locate the location of wheelchair if so required for monitoring their location.

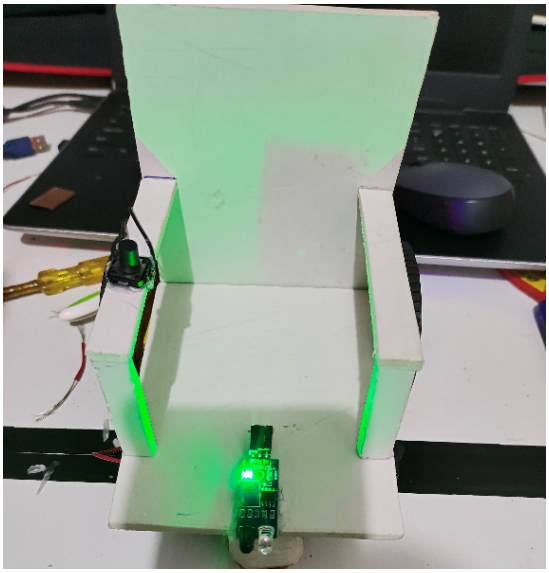
Case 5 :- Obstruction detection:- In our working project we have added feature of objects detection which can avoid any accidents that patients might face during operation.

Case 6 :- Emergency Stop Button:- The project has an emergency Stop Button to Stop the whole project if so required for any injuries for the patient may occur. It has a switch which will Directly cutoff the connection between batteries and controller to avoid accidents.

Case7 :- We have also interfaced a joystick module to move wheelchair without mobile application.

3.4 Hardware pictorial presentation





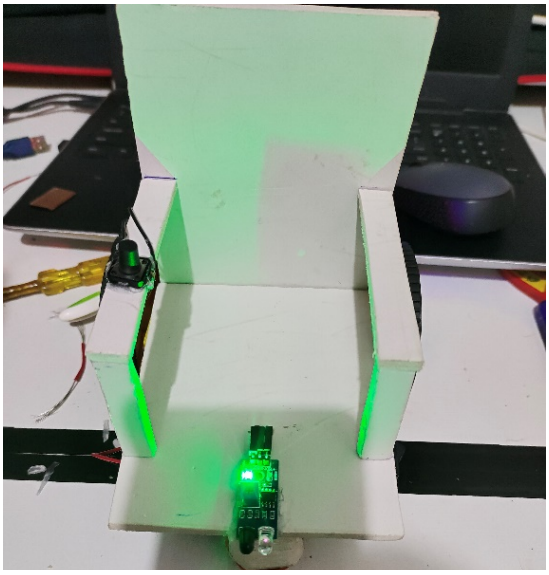


Figure 23 :- Project complete work

3.2 Software Details

3.2.1 Ardunio programming steps

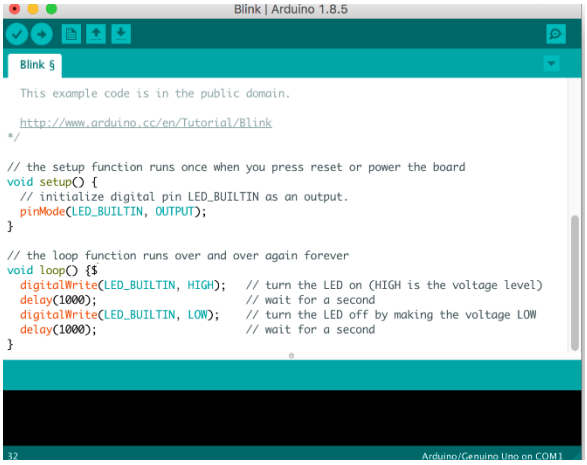


Figure 23 :- Ardunio programming step 1

The programming of an Arduino Mega 2560 can be done with the help of an IDE (Arduino Software), and it supports C-programming language. Here the sketch is the code in the software which is burned within the software and then moved to the Arduino board using a USB cable.

An Arduino mega board includes a boot loader which eliminates an external burner utilization to burn the program code into the Arduino board. Here, the communication of the boot loader can be done using an STK500 protocol.

When we compile as well as burn the Arduino program, then we can detach the USB cable to remove the power supply from the Arduino board. Whenever you propose to use the Arduino board for your project, the power supply can be provided by a power jack otherwise Vin pin of the board.

Download & install the Arduino environment (IDE)

If you just got your Arduino Uno board, you'll first have to install the Arduino IDE (Integrated Development Environment) on another computer. The code is typed into the IDE and sent to the Arduino via a USB cable.

Visit arduino.cc to download the most recent Arduino IDE version for your computer. There are different versions for Mac, Windows, and Linux OS.

- At the download page, click on the "Installer" option for the easiest installation then
- Save the .exe file to your disk drive.
- Open the .exe file.
- Click the button to agree to the licensing agreement
- Decide which components to put in, then click "Next"
- Select which folder to put in the program to, then click "Install"
- Wait for the program to complete installing, then click "Close"

2. Launch the Arduino IDE

After your Arduino IDE software is downloaded, unzip the folder. To do so, double-click on the Arduino shortcut on your Desktopt. The IDE will open up and you'll see the code editor.

3. If needed, install the drivers

If you used the Installer, it'll install drivers automatically as soon as you connect your board.

4. Connect the board to your computer via the USB cable

To power up your board, connect your Arduino board with the pc via USB cable. The green color power LED should glow on the board.

5. Select your board

Next, make sure the software is ready up for your particular Arduino board. Go to the "Tools" computer menu from the menu bar. Select the "Board" option and another menu will appear, where you'll select your Arduino model from the list.

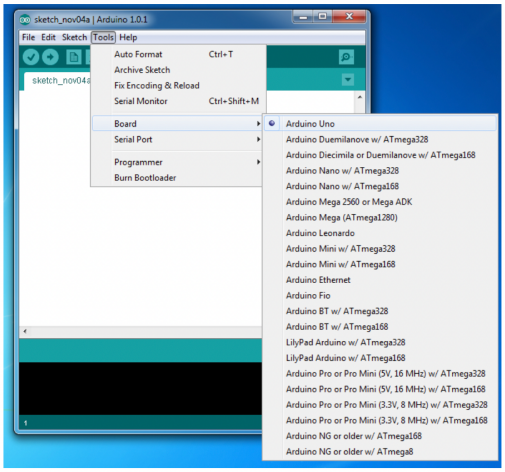


Figure 24 :- Ardunio programming step 2

6. Select your serial port

Select the serial device of the Arduino board. Go to Tools, and then the serial port menu. You might see COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out which port your Arduino board is connected to, disconnect your Arduino board and re-open the menu. The entry that disappears should be the Arduino board. Reconnect the board and choose that serial port.

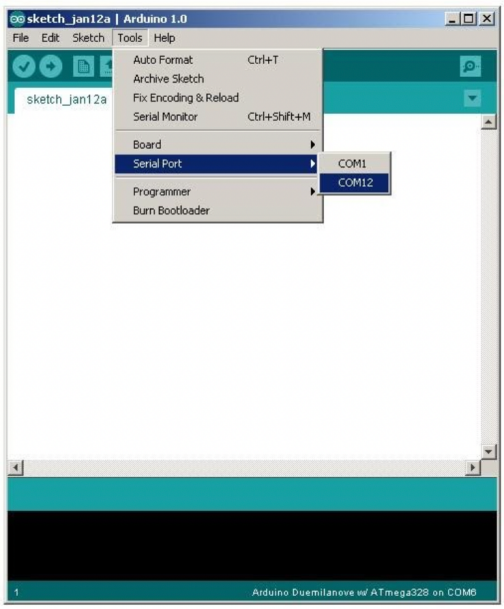


Figure 25 :- Ardunio programming step 3

7. Open the blink example

We'll start with the LED Blink example that comes with the Arduino IDE. Just go to File->Examples->Basics->Blink.

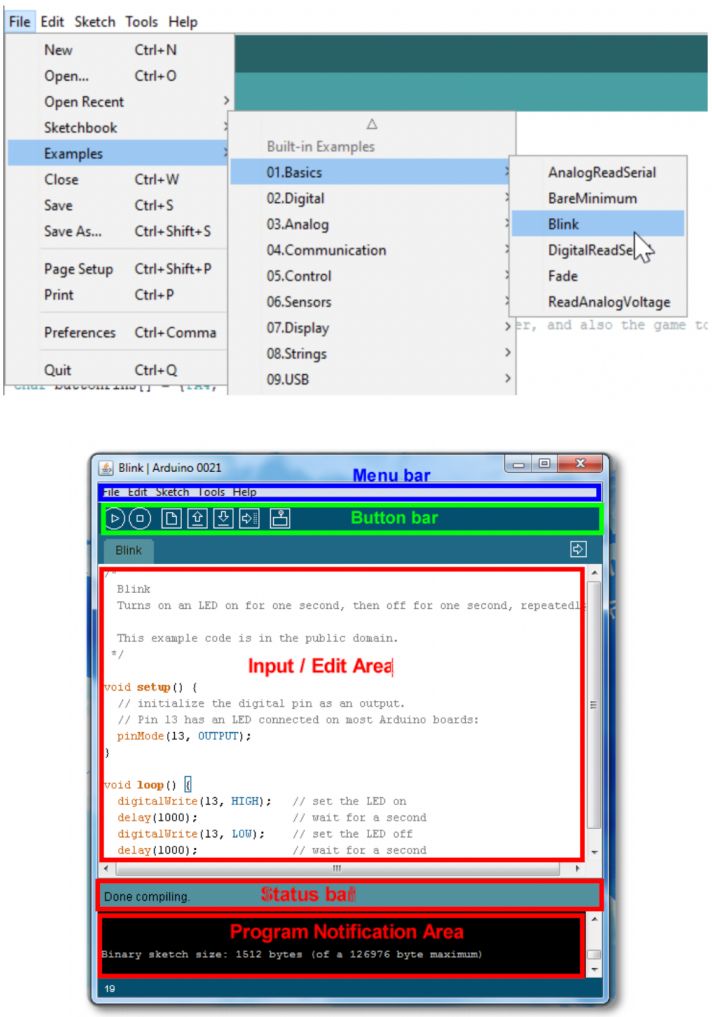


Figure 26 :- Ardunio programming step 4

Here are the few things to keep in mind while writing the code:

- Code is case sensitive
- All the statements must end with a semicolon
- Comments follow a // or begin with /* and end with */

- Void loop() and void setup() are two mandatory functions. The setup section of the code is simply run once when the Arduino board is first turned on or reset. Once the setup is complete, the loop runs over and over. It keeps on running until the board continues to stay powered.
- The status bar shows that the program is compiled or uploaded.
- Program notification area shows error(s) within the code if any.

8. Upload the program

Now it is time to upload your first sketch(code). Confirm the Arduino is plugged in, and the green light is on - therefore the correct board and port is chosen. Select Upload from the Sketch drop-down menu.

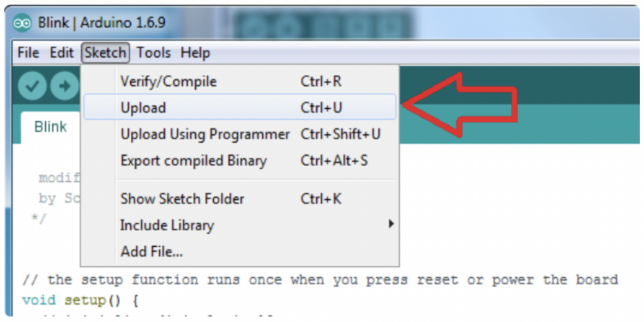
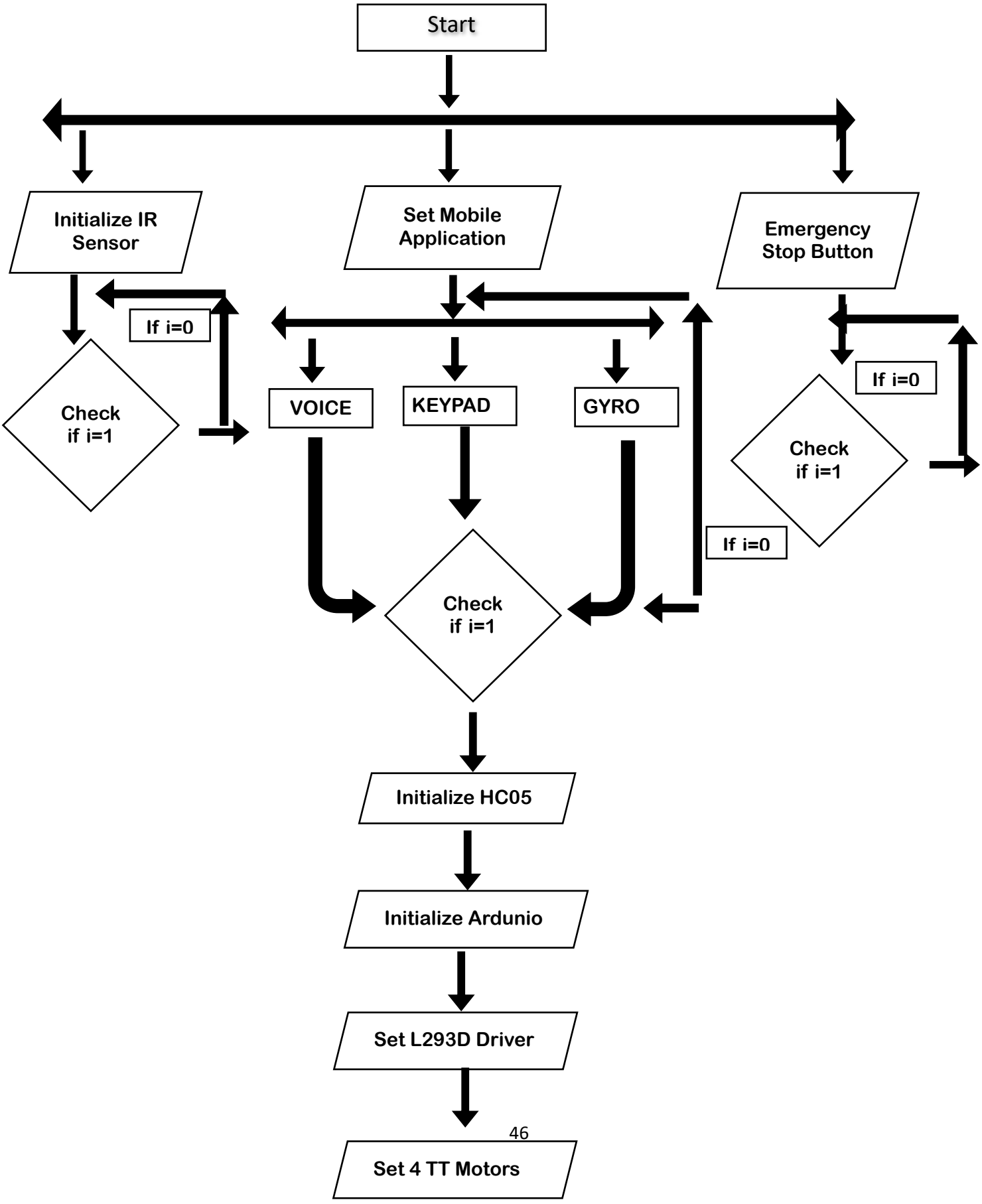


Figure 27 :- Ardunio programming step 5

After a few seconds, you will get this screen, with the message "Done uploading."

3.2.2 Software flow chart



3.2.3 Ardunio programming

```
#define m1 4
#define m2 5
#define m3 6
#define m4 7

#define ir A0

void setup() {

   pinMode(ir, INPUT);
  pinMode(m1, OUTPUT);
 pinMode(m2, OUTPUT);
 pinMode(m3, OUTPUT);
 pinMode(m4, OUTPUT);
 //Start the serial communication
 Serial.begin(9600);
 //Set the motor speeds
//  M1.setSpeed(Speed);
//  M2.setSpeed(Speed);
//  M3.setSpeed(Speed);
//  M4.setSpeed(Speed);
}
void loop() {
 if (analogRead(ir) < 666) {
```

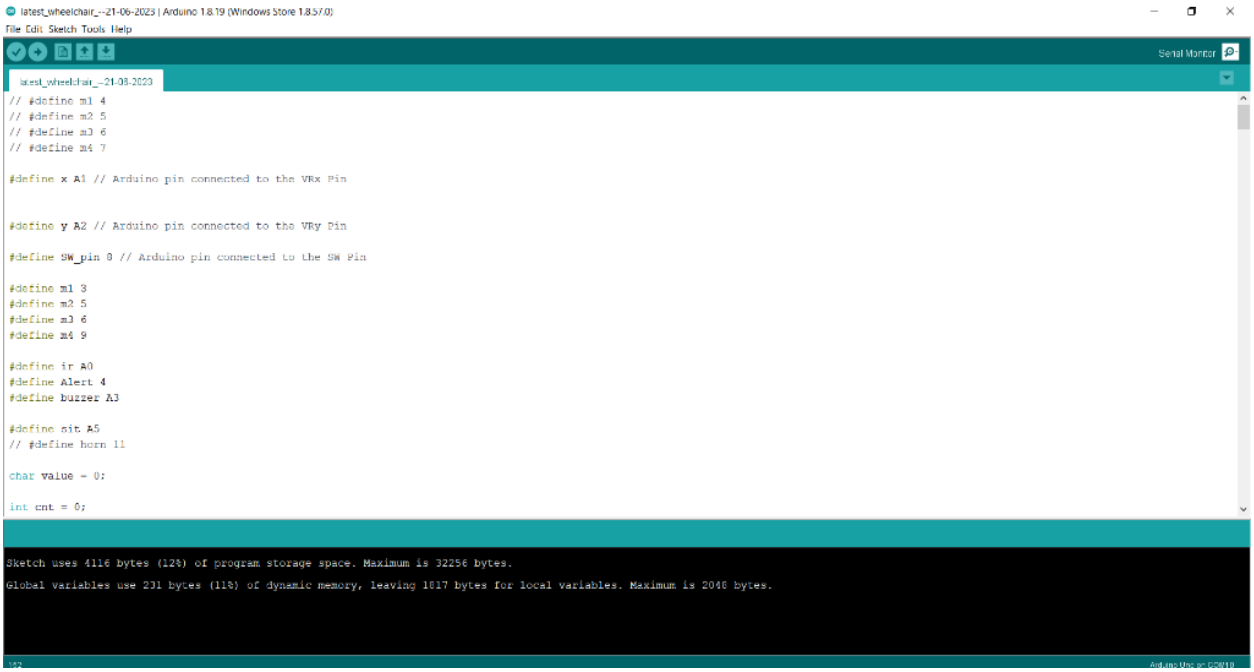
```
    Stop();
   }
  bluetoothControl();//Bluetooth control function

}

void bluetoothControl() {
  //Get the Bluetooth control remote values
  if (Serial.available() > 0) {
    char value = Serial.read();
    Serial.println(value);

    if (value == 'U' && analogRead(ir) >= 666 ) {
      forward();
    } else if (value == 'D') {
      backward();
    } else if (value == 'L') {
      left();
    } else if (value == 'R') {
      right();
    } else if (value == 'S') {
      Stop();
    }

  }
}
```

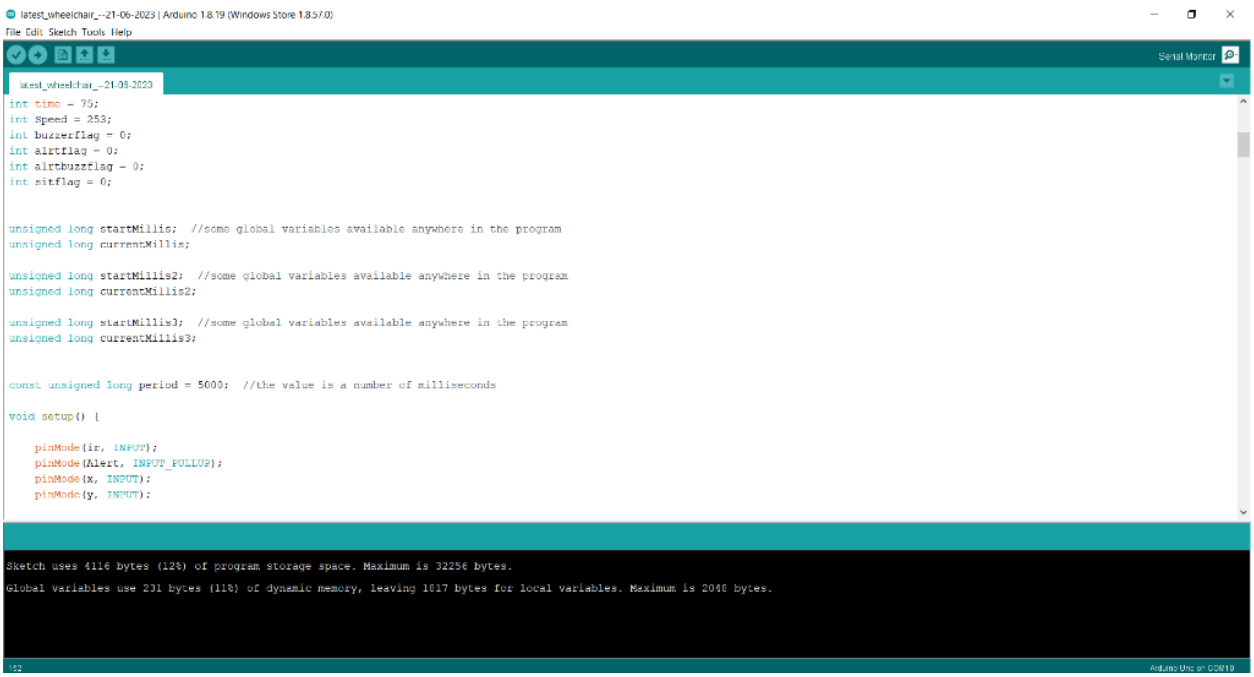
```
/******************Motor functions*****************/
void forward() {
//  M1.run(FORWARD);
//  M2.run(FORWARD);
//  M3.run(FORWARD);
//  M4.run(FORWARD);

   //forward
     digitalWrite(m1, HIGH);
     digitalWrite(m2, LOW);
     digitalWrite(m3, HIGH);
     digitalWrite(m4, LOW);
     Serial.println("Going forward!");
}
void backward() {
//  M1.run(BACKWARD);
//  M2.run(BACKWARD);
//  M3.run(BACKWARD);
//  M4.run(BACKWARD);

   //backward
     digitalWrite(m1, LOW);
     digitalWrite(m2, HIGH);
     digitalWrite(m3, LOW);
     digitalWrite(m4, HIGH);
     Serial.println("Going backward!");
}
```

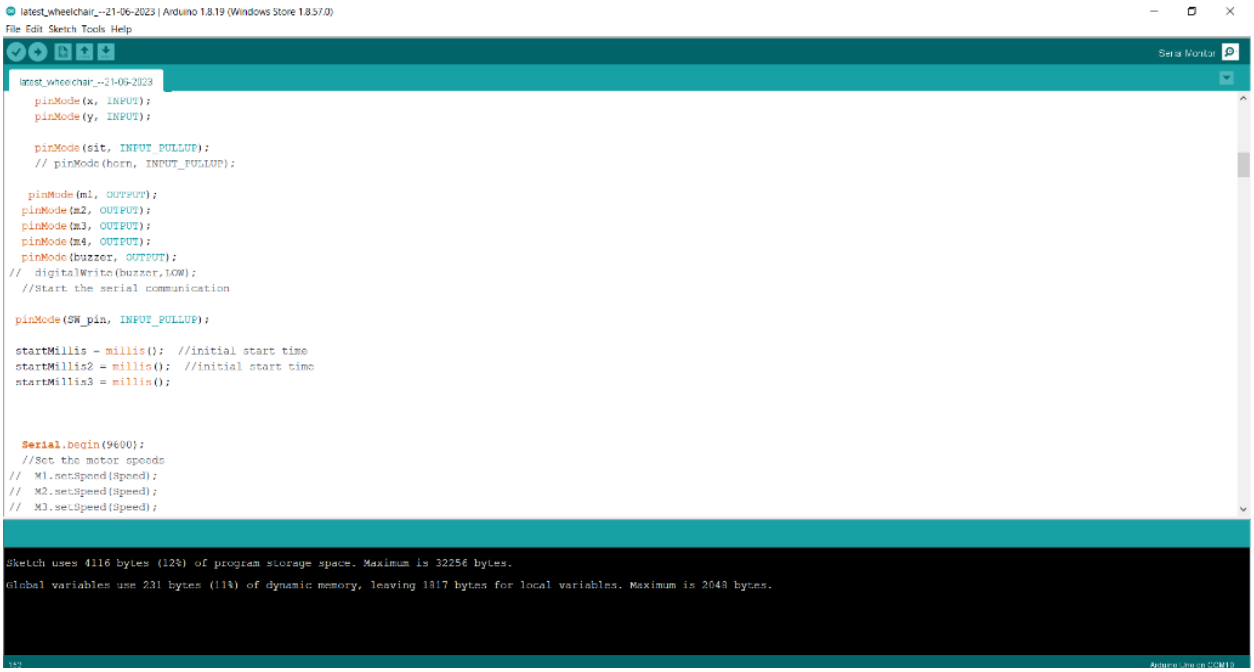
```
void right() {
//  M1.run(FORWARD);
//  M2.run(BACKWARD);
//  M3.run(BACKWARD);
//  M4.run(FORWARD);

   //right
     digitalWrite(m1, HIGH);
     digitalWrite(m2, LOW);
     digitalWrite(m3, LOW);
     digitalWrite(m4, HIGH);
     Serial.println("Going right!");
}
void left() {
//  M1.run(BACKWARD);
//  M2.run(FORWARD);
//  M3.run(FORWARD);
//  M4.run(BACKWARD);

   //left
     digitalWrite(m1, LOW);
     digitalWrite(m2, HIGH);
     digitalWrite(m3, HIGH);
     digitalWrite(m4, LOW);
     Serial.println("Going left!");
}
void Stop() {
```

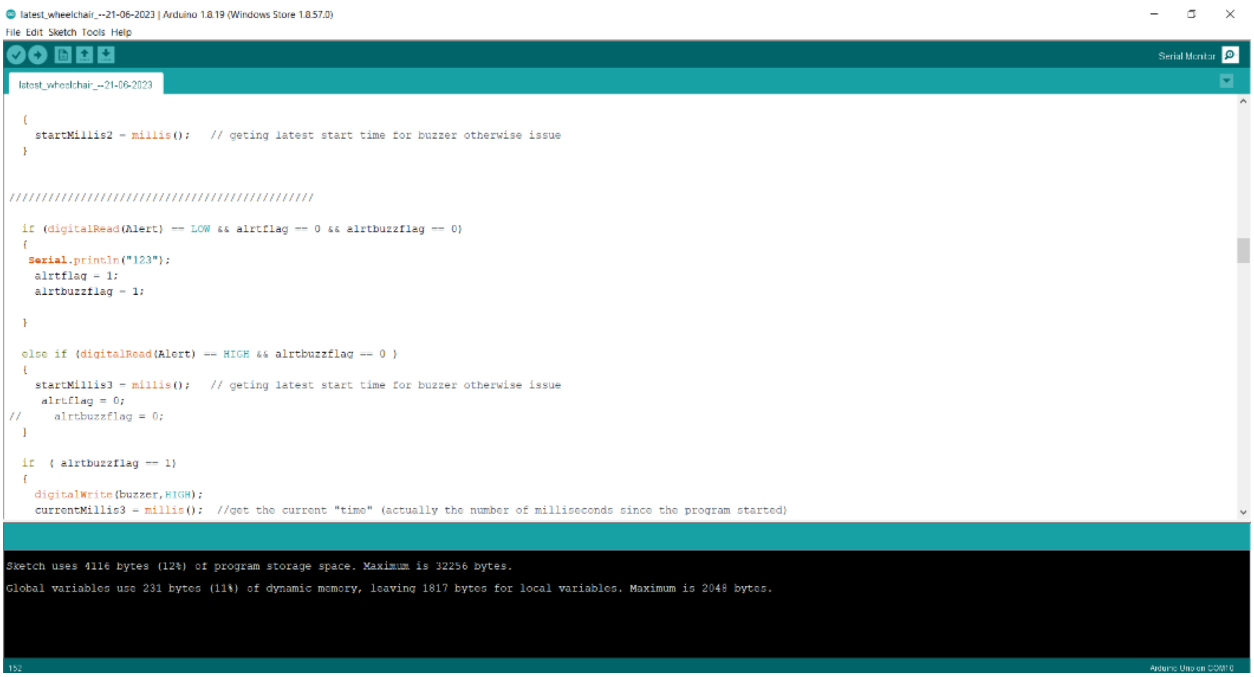
```
//  M1.run(RELEASE);
//  M2.run(RELEASE);
//  M3.run(RELEASE);
//  M4.run(RELEASE);

   //all off
     digitalWrite(m1, LOW);
     digitalWrite(m2, LOW);
     digitalWrite(m3, LOW);
     digitalWrite(m4, LOW);
```

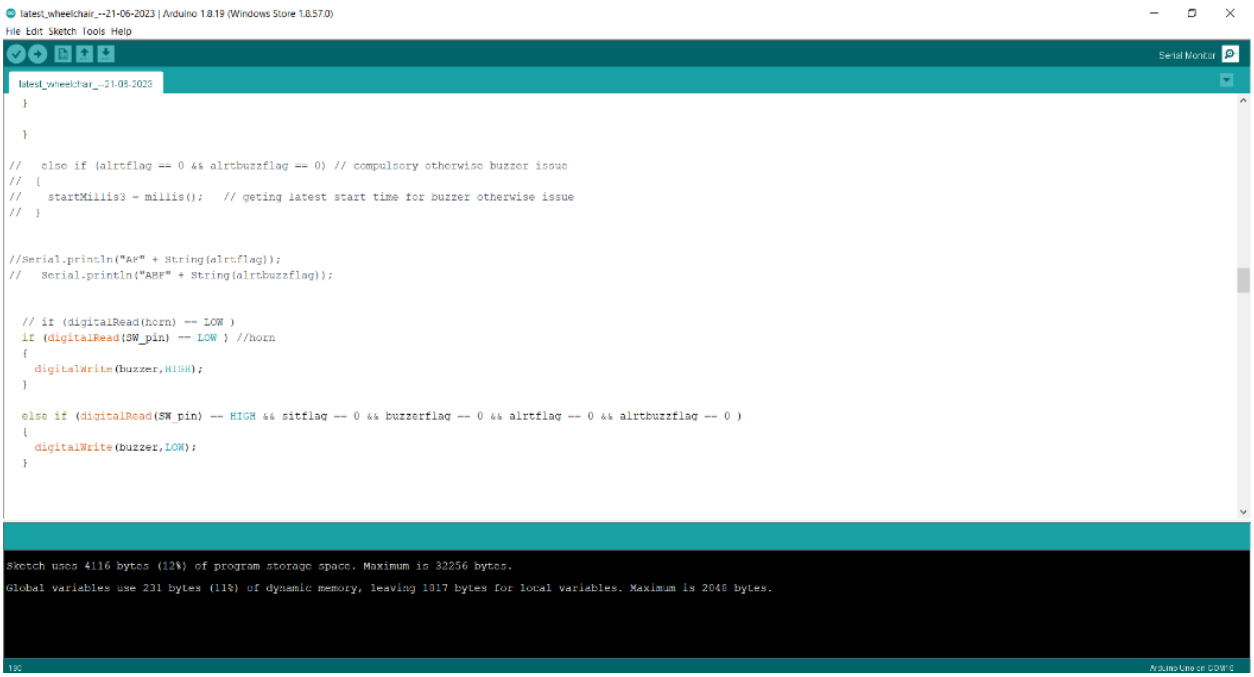
3.2.4 Arduino Programming

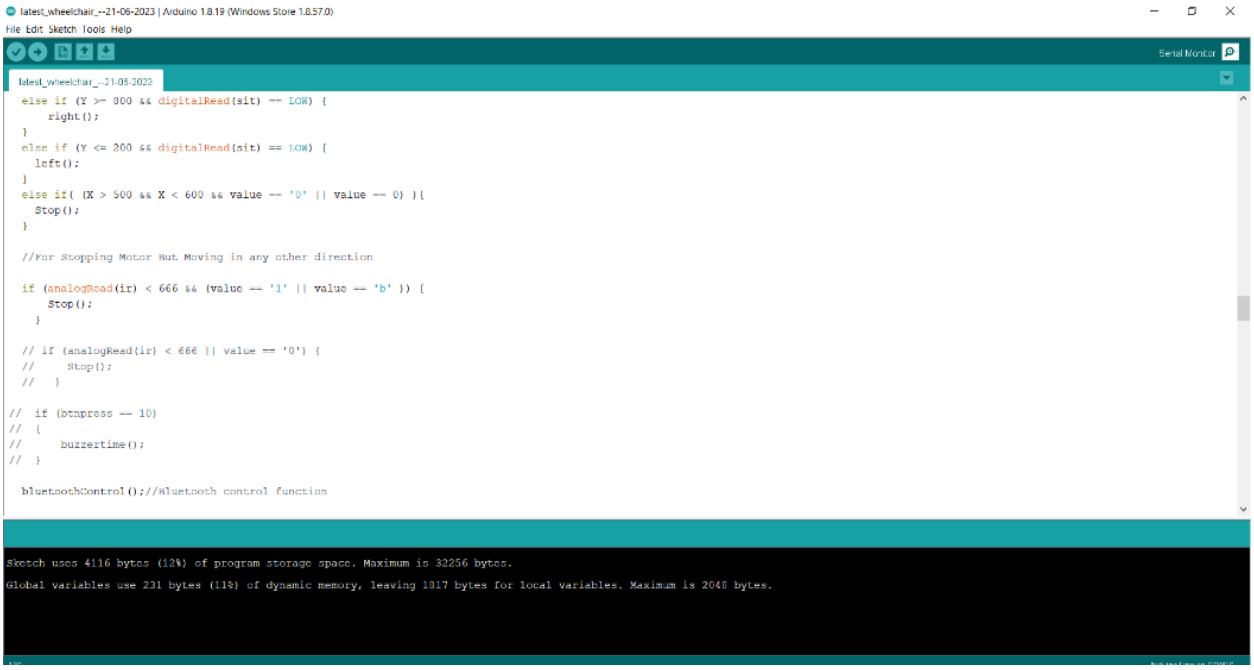

latest_wheelchair_-21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_-21-08-2023
// #define m1 4
// #define m2 5
// #define m3 6
// #define m4 7
#define x A1 // Arduino pin connected to the VRx Pin
#define y A2 // Arduino pin connected to the VRy Pin
#define SW_pin 8 // Arduino pin connected to the SW Pin
#define m1 3
#define m2 5
#define m3 6
#define m4 9
#define ir A0
#define Alert 4
#define buzzer A3
#define sit A5
// #define horn 11
char value = 0;
int cnt = 0;
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.
Arduino Uno on COM19

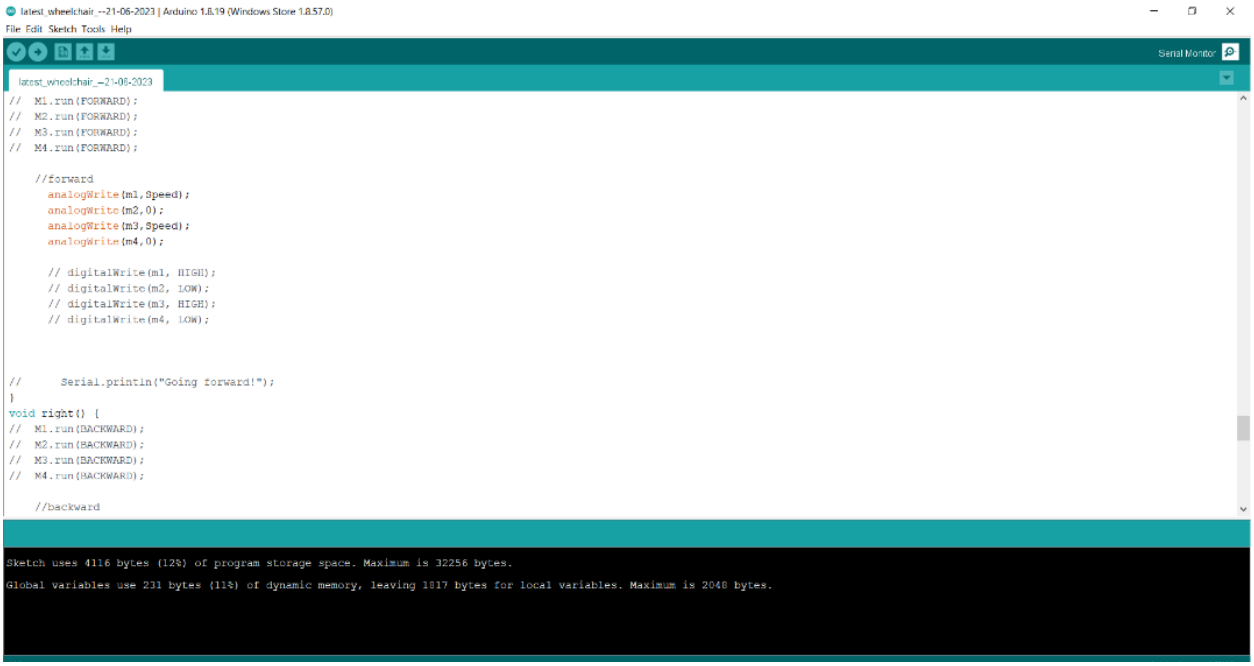

latest_wheelchair_-21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_-21-08-2023
int time = 75;
int Speed = 253;
int buzzerflag = 0;
int alrtflag = 0;
int alrtbuzzflag = 0;
int sitflag = 0;
unsigned long startMillis; //some global variables available anywhere in the program
unsigned long currentMillis;
unsigned long startMillis2; //some global variables available anywhere in the program
unsigned long currentMillis2;
unsigned long startMillis3; //some global variables available anywhere in the program
unsigned long currentMillis3;
const unsigned long period = 5000; //the value is a number of milliseconds
void setup() {
pinMode(ir, INPUT);
pinMode(Alert, INPUT_PULLUP);
pinMode(x, INPUT);
pinMode(y, INPUT);
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.
Arduino Uno on COM19

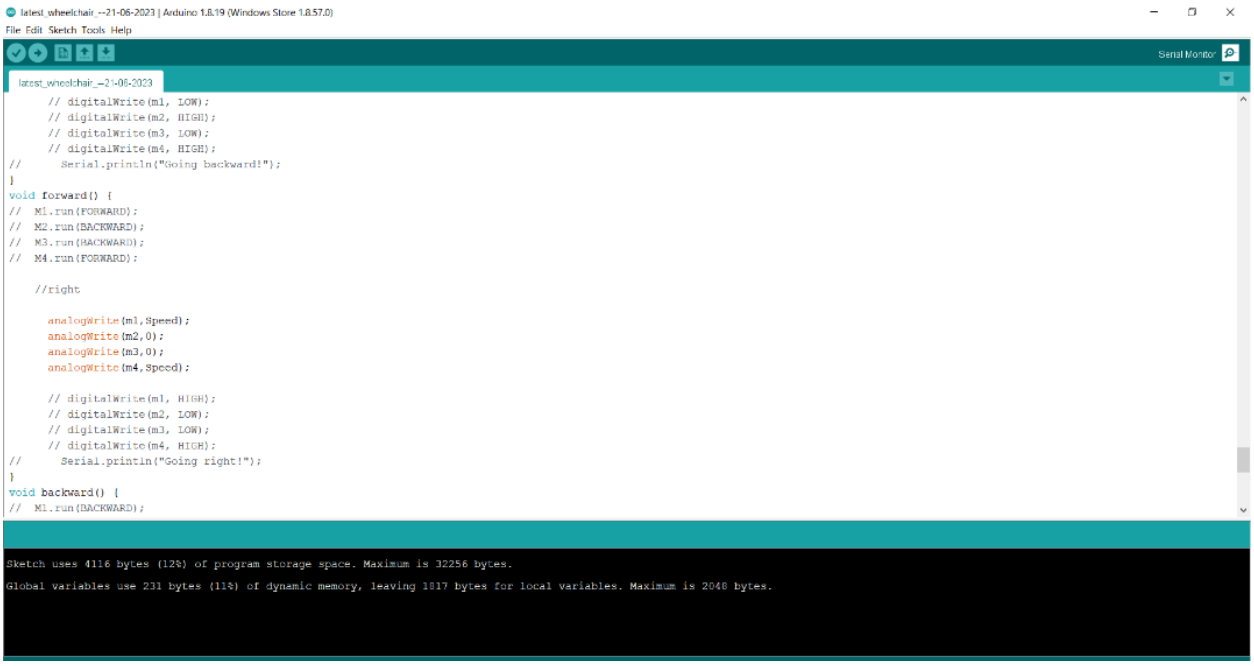

latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
pinMode(x, INPUT);
pinMode(y, INPUT);
pinMode(sit, INPUT_PULLUP);
// pinMode(horn, INPUT_PULLUP);
pinMode(m1, OUTPUT);
pinMode(m2, OUTPUT);
pinMode(m3, OUTPUT);
pinMode(m4, OUTPUT);
pinMode(buzzer, OUTPUT);
// digitalWrite(buzzer,LOW);
//Start the serial communication
pinMode(SW_pin, INPUT_PULLUP);
startMillis = millis(); //initial start time
startMillis2 = millis(); //initial start time
startMillis3 = millis();
Serial.begin(9600);
//Set the motor speeds
// M1.setSpeed(Speed);
// M2.setSpeed(Speed);
// M3.setSpeed(Speed);
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.

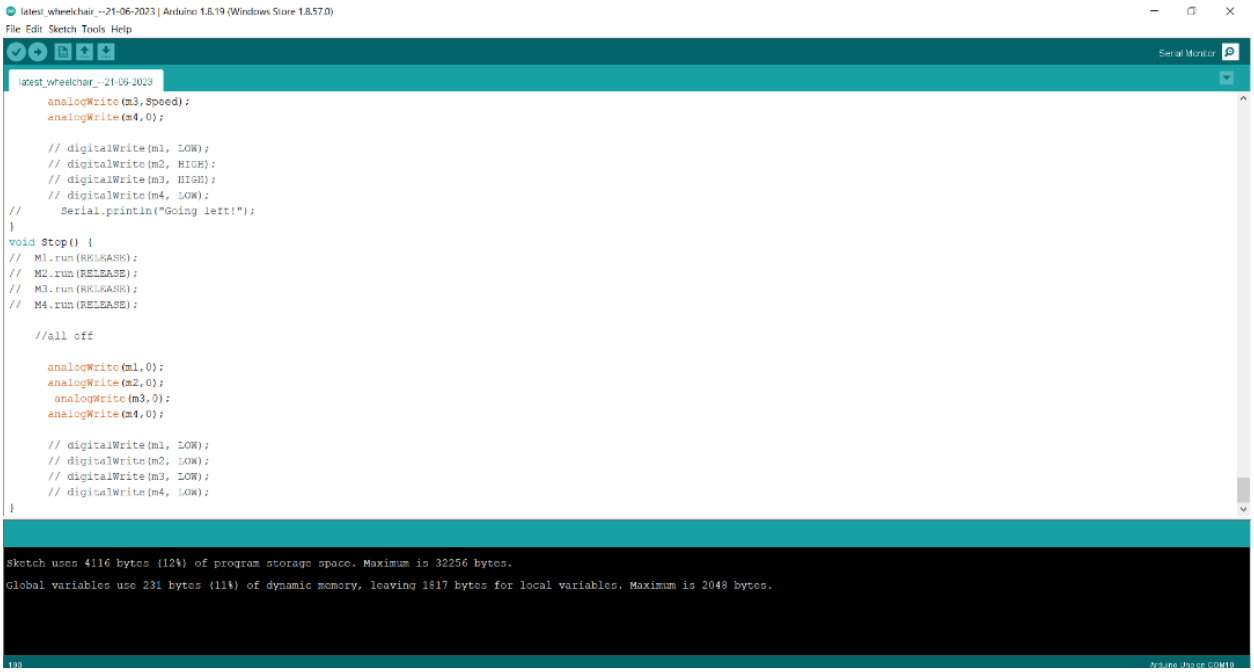

latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
{
startMillis2 = millis(); // geting latest start time for buzzer otherwise issue
}
//
if (digitalRead(Alert) == LOW && alrtflag == 0 && alrtbuzzflag == 0)
{
Serial.println("123");
alrtflag = 1;
alrtbuzzflag = 1;
}
else if (digitalRead(Alert) == HIGH && alrtbuzzflag == 0)
{
startMillis3 = millis(); // geting latest start time for buzzer otherwise issue
alrtflag = 0;
// alrtbuzzflag = 0;
}
if (alrtbuzzflag == 1)
{
digitalWrite(buzzer,HIGH);
currentMillis3 = millis(); //get the current "time" (actually the number of milliseconds since the program started)
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.


latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
}
}
// else if (alrtflag == 0 && alrtbuzzflag == 0) // compulsory otherwise buzzer issue
// {
// startMillis3 = millis(); // geting latest start time for buzzer otherwise issue
// }
//Serial.println("AF" + String(alrtflag));
// Serial.println("ABF" + String(alrtbuzzflag));
// if (digitalRead(horn) == LOW)
if (digitalRead(SW_pin) == LOW) //horn
{
digitalWrite(buzzer,HIGH);
}
else if (digitalRead(SW_pin) == HIGH && sitflag == 0 && buzzerflag == 0 && alrtflag == 0 && alrtbuzzflag == 0)
{
digitalWrite(buzzer,LOW);
}
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.
Arduino Uno on COM10


latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
else if (Y >= 800 && digitalRead(sit) == LOW) {
right();
}
else if (Y <= 200 && digitalRead(sit) == LOW) {
left();
}
else if((X > 500 && X < 600 && value == '0' || value == 0)){
Stop();
}
//For Stopping Motor But Moving in any other direction
if (analogRead(ir) < 666 && (value == '1' || value == 'b')) {
Stop();
}
// if (analogRead(ir) < 666 || value == '0') {
// Stop();
// }
// if (btnpress == 10)
// {
// buzzertime();
// }
bluetoothControl();//Bluetooth control function
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.


latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
// M1.run(FORWARD);
// M2.run(FORWARD);
// M3.run(FORWARD);
// M4.run(FORWARD);
//forward
analogWrite(m1,Speed);
analogWrite(m2,0);
analogWrite(m3,Speed);
analogWrite(m4,0);
// digitalWrite(m1, HIGH);
// digitalWrite(m2, LOW);
// digitalWrite(m3, HIGH);
// digitalWrite(m4, LOW);
// Serial.println("Going forward!");
}
void right() {
// M1.run(BACKWARD);
// M2.run(BACKWARD);
// M3.run(BACKWARD);
// M4.run(BACKWARD);
//backward
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.


latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
// digitalWrite(m1, LOW);
// digitalWrite(m2, HIGH);
// digitalWrite(m3, LOW);
// digitalWrite(m4, HIGH);
// Serial.println("Going backward!");
}
void forward() {
// M1.run(FORWARD);
// M2.run(BACKWARD);
// M3.run(BACKWARD);
// M4.run(FORWARD);
//right
analogWrite(m1,Speed);
analogWrite(m2,0);
analogWrite(m3,0);
analogWrite(m4,Speed);
// digitalWrite(m1, HIGH);
// digitalWrite(m2, LOW);
// digitalWrite(m3, LOW);
// digitalWrite(m4, HIGH);
// Serial.println("Going right!");
}
void backward() {
// M1.run(BACKWARD);
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.


latest_wheelchair_--21-06-2023 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
latest_wheelchair_--21-06-2023
analogWrite(m3,Speed);
analogWrite(m4,0);
// digitalWrite(m1, LOW);
// digitalWrite(m2, HIGH);
// digitalWrite(m3, HIGH);
// digitalWrite(m4, LOW);
// Serial.println("Going left!");
}
void Stop() {
// M1.run(RELEASE);
// M2.run(RELEASE);
// M3.run(RELEASE);
// M4.run(RELEASE);
//all off
analogWrite(m1,0);
analogWrite(m2,0);
analogWrite(m3,0);
analogWrite(m4,0);
// digitalWrite(m1, LOW);
// digitalWrite(m2, LOW);
// digitalWrite(m3, LOW);
// digitalWrite(m4, LOW);
}
Sketch uses 4116 bytes (12%) of program storage space. Maximum is 32256 bytes.
Global variables use 231 bytes (11%) of dynamic memory, leaving 1817 bytes for local variables. Maximum is 2048 bytes.

3.2.4 MIT App Inventor

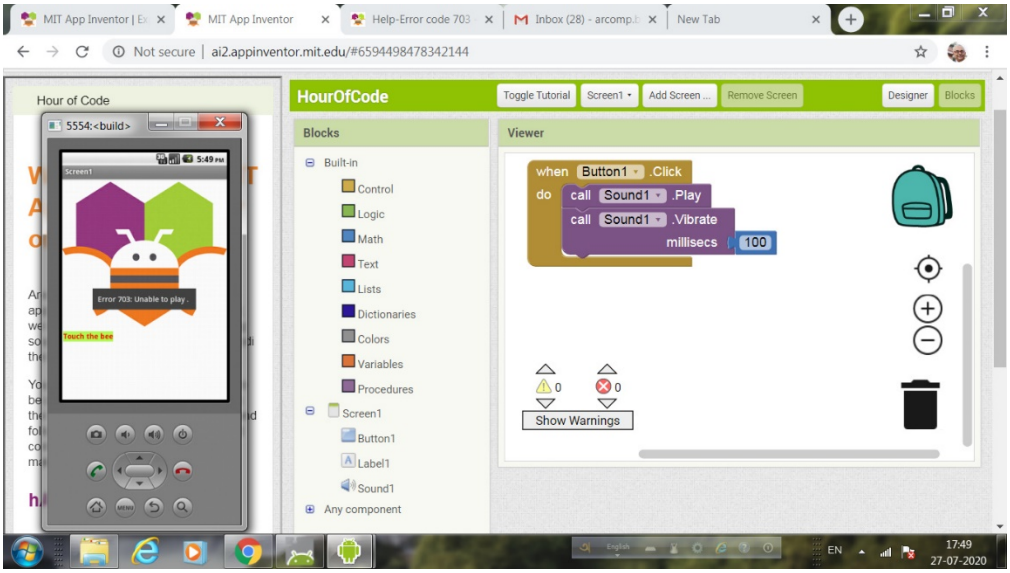


Figure 28 :- MIT App inventor interface

In recent years the internet has reached a large number of people across the world. And this has motivated all the people who own some businesses or people who want to connect to a large number of people to use and make mobile/laptop applications. These applications provide the customer with a way of easily exploring the content of the provider and it helps the provider with an ample opportunity to present their content more efficiently then they could when they were using websites.

But not all people can make the applications. They require some medium to get their business on an application or an app. One way is to hire app developers. It can prove to be costly for some new business setters or the businessmen can be living in some remote part of the country where app developers are not present. Hence, to make app development easier MIT provides us with the MIT app

inventor. This is a platform that makes app development easy for anyone who knows to code or not.

Steps to use MIT App Inventor

Step 1: Open a Gmail account in case you don't have one.

Step 2: Open the link https://appinventor.mit.edu/ and log in to your Gmail account.

Step 3: You need to install the App Inventor Companion App(MIT AI2 Companion) on our mobile device that helps in live testing of our application.

Step 4: We need to connect both mobile devices & laptops/desktop should be connected to the same WiFi network.

Step 5: To start the app-building click on "Start New Project"

Step 6: To connect your mobile device, choose "Connect" and "AI Companion" from the top

menu.

Step 7: Now to connect the MIT AI2 App on your device and desktop/laptop scan the QR code or

type the 6 digit code which is appearing on your PC screen.

Step 8: Now you can see the app you are building on your device.

5.3.2 Benefits of MIT App Inventor

● Everything is done through a select and drop manner. This means we can select a particular chunk of code and drop in our code. Hence, no typing.

● Easy to test your app. We can check the app developed on desktop or laptop with the

app inventor application on our mobile phones.

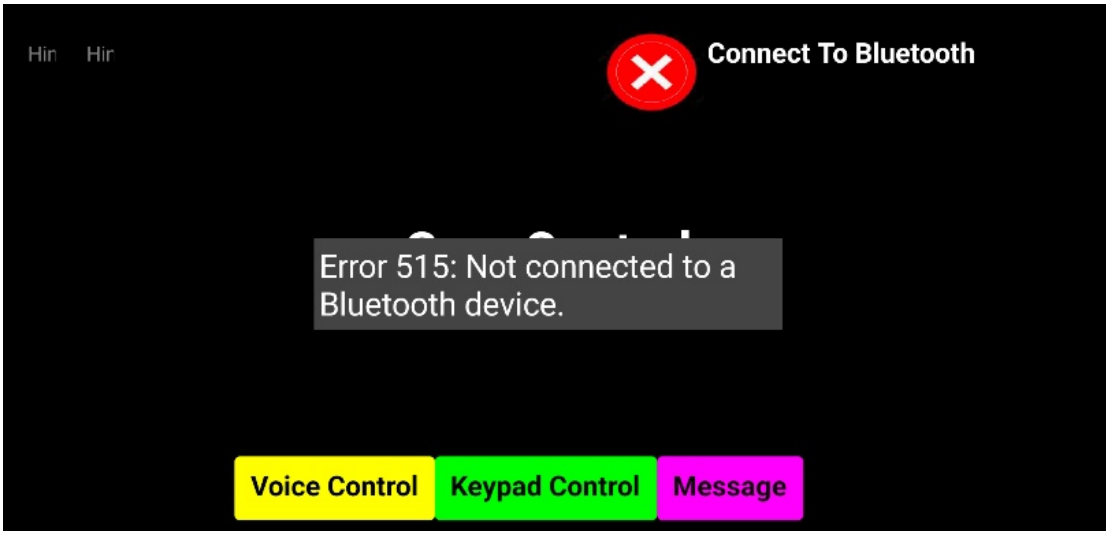
● MIT provides the user with some basic lessons which help in building that apps and

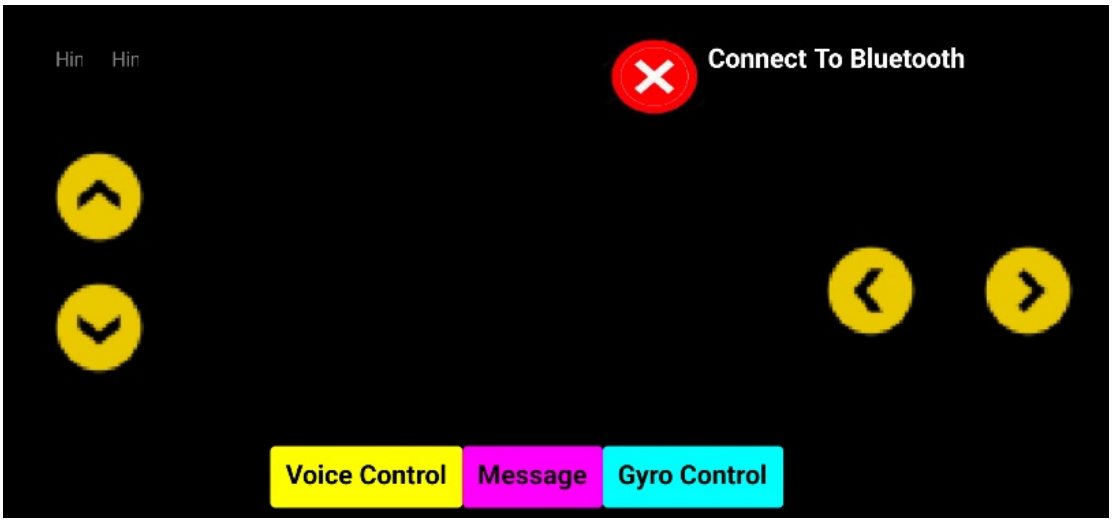
that helps in a proper understanding of how the MIT app inventor platform works for

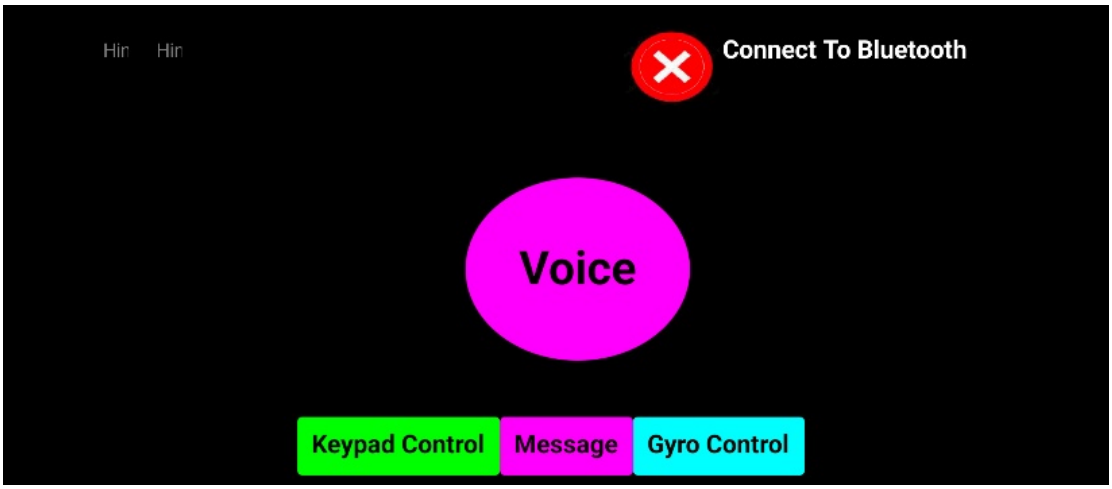
the user.

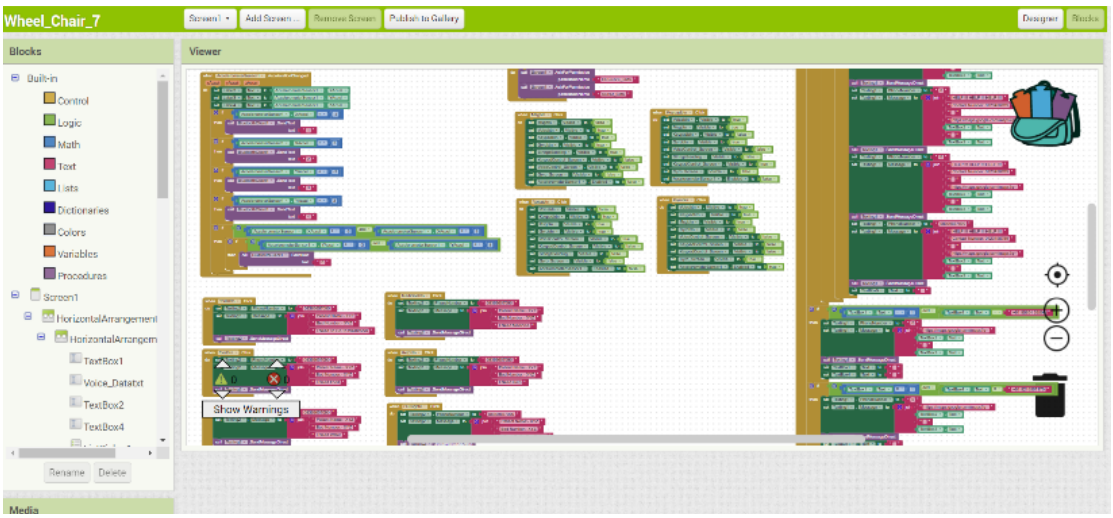
● Power of native apps with a simple UI

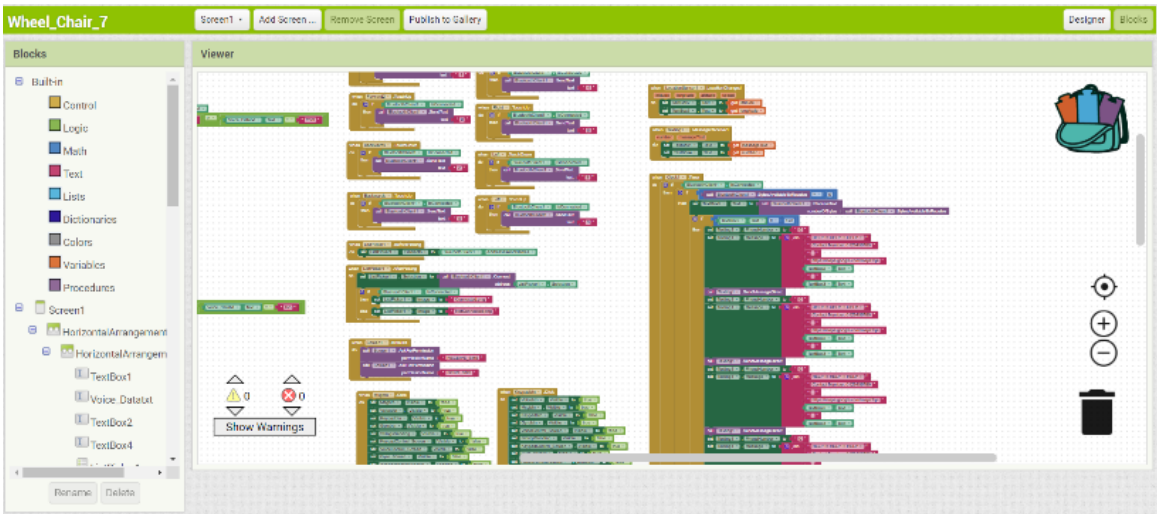
3.2.5 Our Application Interface

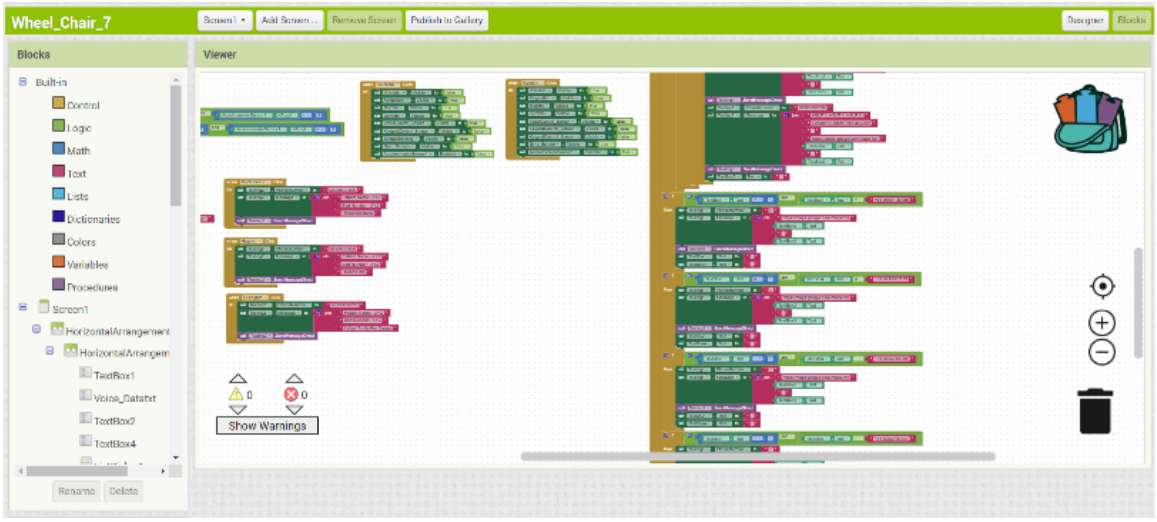


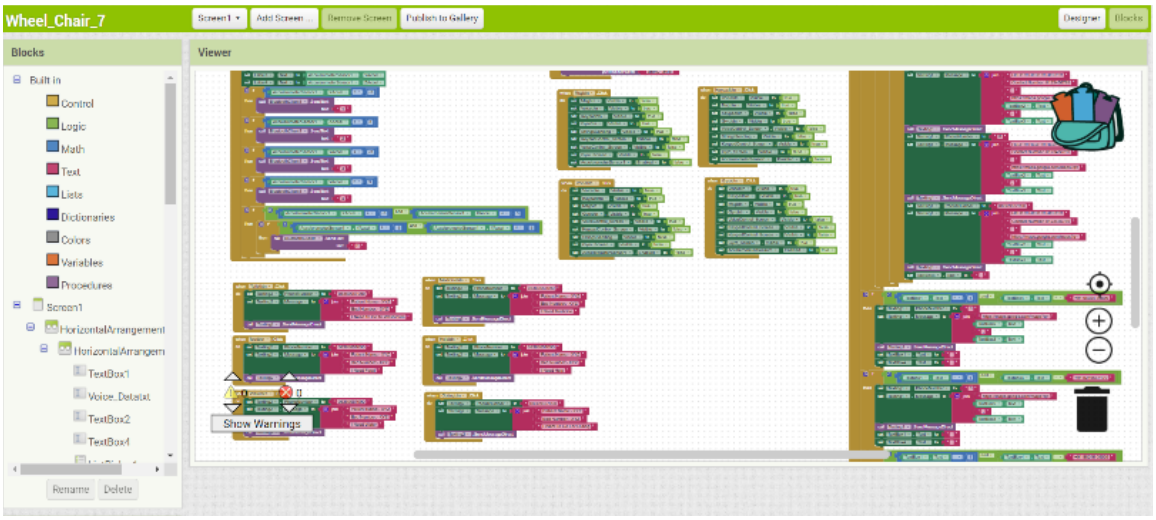


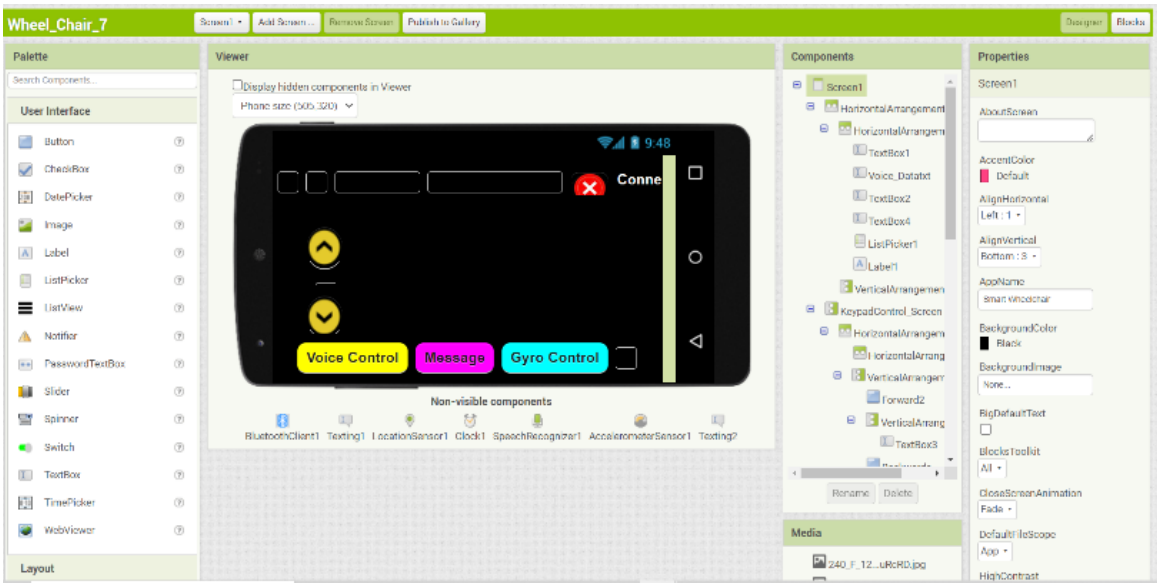

Hin Hin
Connect To Bluetooth
Voice
Keypad Control
Message
Gyro Control


Wheel_Chair_7
Blocks
Viewer
Show Warnings


Wheel_Chair_7
Blocks
Viewer
Show Warnings


Wheel_Chair_7
Blocks
Viewer
Show Warnings


Wheel_Chair_7
Blocks
Viewer
Show Warnings


Wheel_Chair_7
Palette
Viewer
Components
Properties
Voice Control
Message
Gyro Control
Non-visible components

CHAPTER 4:- CONCLUSIONS

Chapter 4
Conclusions

Advantage:-

1) Simple Usage for patients.
2) There are so many functions for a person handling this wheelchair.
3)Use of Pic Controller will reduce the Size of project
4)Being a prototype project looks complex but when will be implemented at last scale will definitely be more cost effective and small in size

Disadvantage:-

1) Prototype Size and Cost is high
2) Coding Complexity will increase with more Upgradeability.
3) Some times one function interrupts other movements. Which can be avoided in further upgrades.

Application:-

1) This is a smart wheelchair for helping physically handicap people.
1) This project can be used by patients with multiple disabilities.

CHAPTER 5:- REFERENCES

Chapter 5

References

for differently abled. 4th IEEE International Conference on Computing, Communication and Networking Technologies, pp. 1-6, India (2013).

2. Nutthanan Wanluk, Sarinporn Visitsattapongse, Aniwat Juhong, C. Pintavirooj.: Smart wheelchair based on eye tracking. 9th IEEE Biomedical Engineering International Conference (BMEiCON), 2016

3. Aruna C., Dhivya P., Malini M., Gopu G.: Voice recognition and touch screen control based wheelchair for paraplegic persons. IEEE International

Conference on Green Computing Communication and Electrical Engineering, P.1-5, India (2014)

4. Romil Chauhan, Yash Jain, Harsh Agarwal, Abhijit Patil.:Study of Implementation of Voice Controlled Wheelchair. 3rd International Conference on Advanced Computing and Communication Systems (ICACCS-2016), 2016.

5. Census of India 2011 "Data on Disability"

6. World Health Organization "World report on Disability".

7. B. Yukesekkaya, et al.: A GSM, Internet and Speech Controlled Wireless Interactive Home Automation System. IEEE Transactions on Consumer Electronics, pp. 837-843. IEEE Press, New York (2006)

8. Nishimori M., Saitoh T., Konishi R.: Voice controlled intelligent wheelchair. IEEE conference, pp.336-340, Japan (2007)

9. Asakawa T., Nishihara K., Yoshidome T.: Experiment on operating methods of an electric wheelchair for a system of detecting position and direction. IEEE International Conference on Robotics and Biomimetics, pp. 1260-1265, China (2007)

10. Simpson R.C., Levine S.P.:Voice control of a power wheelchair. IEEE Transaction on Neural Systems and Rehabilitation Engineering, vol.10 (2), pp.122-125, IEEE Press, New York (2002)

11. R. C. Simpson.:Smart wheelchairs: a literature review. Journal of Rehabilitation Research and Development (2005)

12. Sarangi P. Parikh, Valdir Grassi Jr., Vijay Kumar, Jun Okamoto Jr.: Integrating Human Inputs with Autonomous Behaviors on an Intelligent Wheelchair Platform. IEEE Computer Society, vol. 22(2), pp. 33-41, IEEE Press, New York (2007)