CHAPTER 1

ABSTRACT

Accelerometer works on MEMS (Micro-Electro Mechanical System) and provide the three axis data for purposeful application. A robot controlled by accelerometer can be handy to transport small goods or can be useful for robot guidance. The smart phone controlling robot using its internal accelerometer can be used to guide small transport devices like “luggage follow transport”system. Since modern Smartphone are equipped with an in-built accelerometer, they can be easy to implement, cheap to provide and the output will be more intuitive. The design and implementation of a wireless smart phone controlled Robot using Arduino processor and an Android operated application to control the movement via Bluetooth with minimal, and cheap hardware requirements. The system can be broadly classified into two components: the hardwareand the software part.

The integration of more and more functionality into the human machine interface (HMI) of vehicles increases the complexity of device handling. Thus optimal use of different human sensory channels is an approach to simplify the interaction with in-car devices. Using this idea, a car-robot can be implemented whose navigation can be done wirelessly with the help of an Arduino UNO and NANO Micro-controllers.

CHAPTER 2
INTRODUCTION

2.1 Introduction:

The current emerging technology in the field of science is Robotics. It is the new emerging booming field of great use to people in the coming years. These days a number of wireless robots are being developed and put to various applications and uses. In order to enhance the contribution of robot in our daily lives we need to find an effective way of communicating with robots. For this purpose, there have been certain developments in area of human-machine interaction. One common form of communication is Gestures that are not only limited to face, body and fingers but also with smart phones. However, gestures have been an important means of communication in the physical world from ancient times, even before the invention of any language. In this era of machines taking control of every complex works. Interactions with machines have become more important than ever.

Robots are classified into two types: Autonomous robots like Line sensing or edge sensing robots, and Remote controlled robots like Gesture controlled Robots. A gesture is a form of communication in a non-verbal manner by using visible body movements or actions conveying messages. There are several ways to capture a human gesture that a machine would be able to understand. The gesture can be captured using a camera, or a data glove. Gestures can also be captured via Bluetooth or infrared waves, Acoustic, Tactile, optical or motion technological means. The embedded systems designed for specific control functions can be optimized to reduce the size and cost of the device, and increase the reliability and performance. With the approch of Smartphone and other modern technologies, operating machines have become more flexible. The Smartphone are equipped with in-built accelerometer which may be used for gesture recognition and such other tasks. Moreover, the Android OS is gaining significant popularity in the world of Smartphone due to its open architecture. Android platform is being used in the development of numerous applications for cell-phones. Apart from hand gesture recognition, emotional gesture recognition from face is also done in some cases.

There are two types of gestures used in gesture recognition: Online gestures and Offline gestures. In Online gestures, direct manipulations like rotation and scaling are done. In Offline gestures, the processing is done only after the user interacts with the object. Gesture technologies are applied in several fields like in Augmented Reality, Socially assistive Robots, recognition of sign languages, emotion detection from facial expressions, Virtual mouse or keyboard, recognition of sign languages, remote control, etc.

There are various modes of communication between the microcontroller of the robot and the Smartphone. However, the popularly used means of communication is done via RF (Radio Frequency), Bluetooth or Wi-Fi. Using RF limits the distance from which the robot can be controlled. Using Wi-Fi increases the overall cost for setup. So, the robot has been built with Bluetooth which has intermediate range of distance covered and cost between RF and Wi-Fi.

An android operated phone is incorporated as an accelerometer and the Arduino microcontroller is incorporated in the robot for the main computation and the main communication between all the modules. Then there is a motor driver that deals with the computation and functioning of the motors to turn the wheels essential for the movement of the robot. Last, but not least, a Bluetooth module is incorporated in the robot that serves as the means of receiving the data from the Smartphone which is processed using Arduino to detect the direction of movement of the user's hand and move the robot accordingly. The prime aim of the design is that as the user moves his hand in some direction, the robot moves in the same direction as well. In other words, the robot is solely controlled by the hand movements and gestures of the user.

2.2 Aim:

The aim is to develop, control and program a robot with gestures and assure high level of abstraction, cheap and minimal hardware and a simplified robot programming.

2.3 Background Information:

There is an exponential development in the field of robotics, and controlling robots through human gestures have been the topic of research for as long time. With the implementation of gestures to control robots, there have been several methodologies to perform the action. Some of the related works are being described in this section:

- **Light-based Gesture Recognition:** Light or illumination tracking and controlling robots with light sensors are being done in a lot of cases. Such robots are autonomous in nature. Generally, there are some light sensors associated with the robot. The sensors send some rays of light and track them as they gets absorbed in the surface or reflected back to it. According to this, the robot can be line-sensing robots where it is made to follow a black or a white path autonomously.

- **Vision-based Gesture Recognition:** Several robots are designed to be controlled by vision-based gestures. In such robots, there are generally, some cameras as the sensor, which also acts as a interface to control the robot with some manipulators. The input gesture can be some patterns, movements of hands, color tracking, face recognition, finger tracking, or some templates. They are also used in ball tracking and Robo-football games where the robots play the traditional game of football by tracking the movement of the ball. Though it has paved a way for advanced robot operations, but it is affected by factors such as illumination, foggy weather, background lights, etc.
- **Motion-based Gesture Recognition:** The motions can be used to control a robot. This is generally done by incorporating an accelerometer to control the robot wirelessly. This can also be done using sensors. This method is beneficial over other methods in the sense that it can interact with machines naturally without being intruded by the factors that affects the mechanical devices.

- **Sixth Sense Technology:** The Sixth sense technology begins in 1990 by Steve Mann who implemented a wearable computing device via neck- projector or head-mounted projector coupled with a camera. Later, following his idea, Pranav Mistry, a young research scientist at MIT at that time came up with new applications of this technology. Pranav Mistry came up with the name “Sixth Sense Technology” and has since been named Wear Ur World (WUW). This technology applies all of the techniques mentioned above and designing applications that give an intuitive output with the connection of internet.

CHAPTER 3

TECHNICAL REQUIREMENTS

The components used in making accelerometer controlled robot are:

- Arduino UNO.
- Software.
- Bluetooth module HC-05.
- Robot chassis.
- DC motors.
- Wheels.
- L293D IC.
- Bread board.
- Battery 12volts.
- Connecting wires.
- Jumpers.

3.1 Arduino UNO:

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control both physically and digitally. Its products are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form or as do-it-yourself (DIY) kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or breadboards (shields) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

Most Arduino boards consist of an Atmel 8-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012. The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed shields. Multiple and possibly stacked shields may be individually addressable via an I²C serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions. Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Arduino UNO is the optiboot bootloader. Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between RS-232 logic levels and transistor–transistor logic (TTL) level signals.

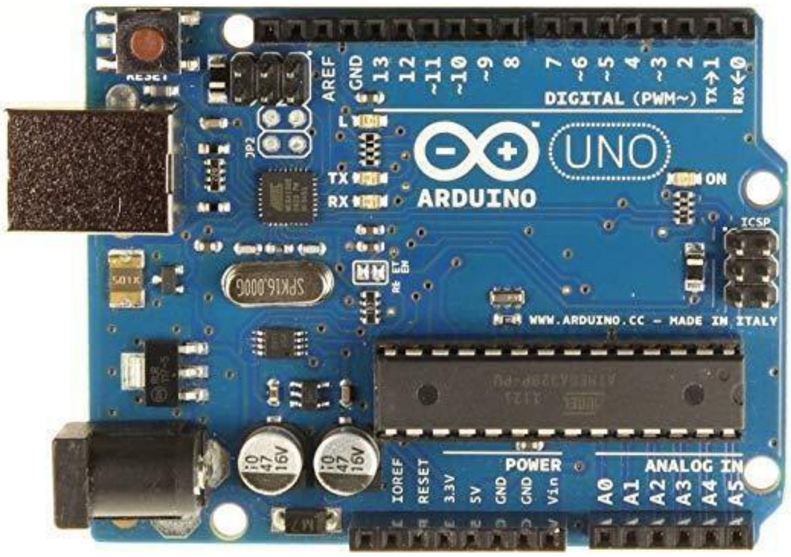

Figure 3.1 Arduino board

Current Arduino boards are programmed via Universal Serial Bus (USB), implemented using USB-to-serial adapter chips such as the FTDI FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Board UNO, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods. When used with traditional microcontroller tools, instead of the Arduino IDE, standard AVR in-system programming (ISP) programming is used.

The board has the following new features:

- pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

3.1.1 Specifications:

3.1.1.1 Microcontroller = ATmega328

3.1.1.2 Operating Voltage = 5V

3.1.1.3 Input Voltage = 7-12V

3.1.1.4 Input Voltage = 6-20V

3.1.1.5 Digital I/O Pins = 14

3.1.1.6 Analog Input Pins = 6

3.1.1.7 DC Current per I/O Pin = 40 mA

3.1.1.8 DC Current for 3.3V Pin = 50 mA

3.1.1.9 Flash Memory = 32 KB

3.1.1.10 SRAM = 2 KB

3.1.1.11 EEPROM = 1 KB (ATmega328)

3.1.1.12 Clock Speed 16 MHz

3.1.2 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. 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. A Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328P also supports I2C and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

3.1.3 Arduino Uno to ATmega328 Pin Mapping:

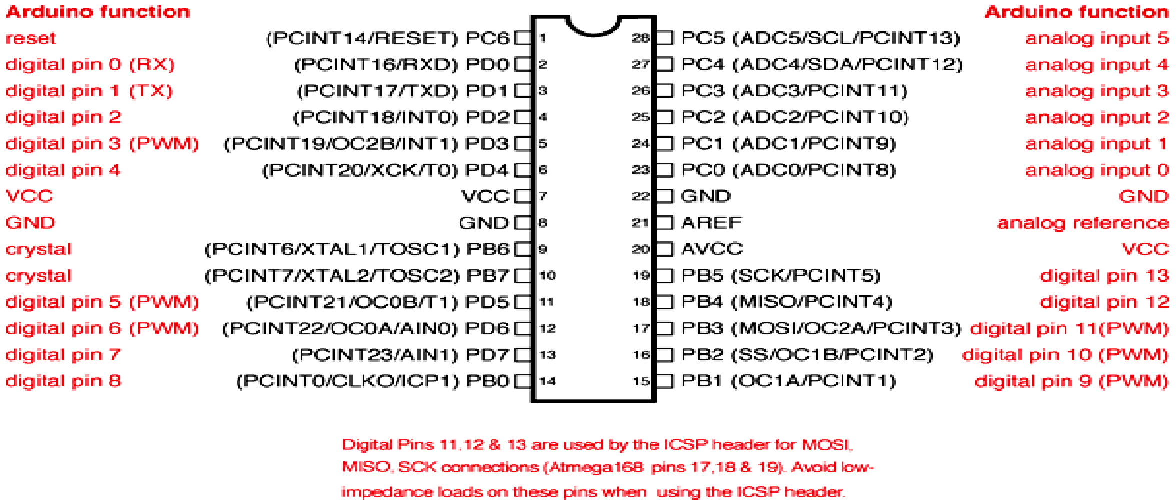

Figure 3.2 AT mega328 pin mapping

3.2 SOFTWARE:

A program for Arduino hardware may be written in any programming language with compilers that produce binary machine code for the target processor. Atmel provides a development environment for their 8-bit AVR and 32-bit ARM Cortex-M based microcontrollers: AVR Studio and Atmel Studio.

3.2.1 Arduino IDE:

The Arduino integrated development environment (IDE) is a cross-platform application for Windows, macOS, Linux. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution.

3.2.2 Arduino IDE Initial Setup:

This is the Arduino IDE once it’s been opened. It opens into a blank sketch where you can start programming immediately. First, we should configure the board and port settings to allow us to upload code. Connect your Arduino board to the PC via the USB cable.

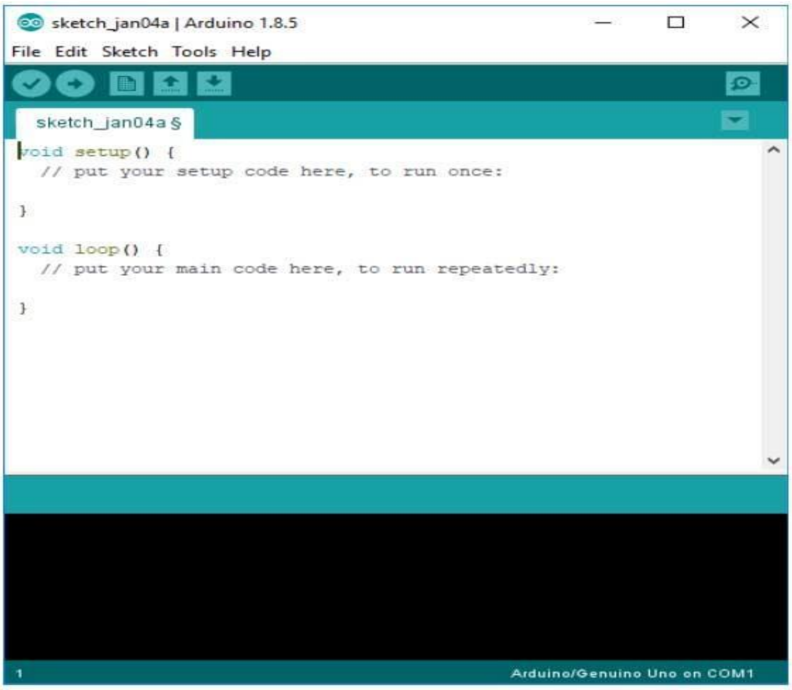

Figure 3.3 Arduino IDE Default Window

3.2.3 Arduino IDE Board Setup:

We have to tell the Arduino IDE what board you are uploading to. Select the **Tools** pulldown menu and go to Board. This list is populated by default with the currently available Arduino Boards that are developed by Arduino. If we are using an Uno or an Uno-Compatible Clone (ex. Funduino, SainSmart, IEIK, etc.), select Arduino Uno or if we are using another board/clone, select that board.

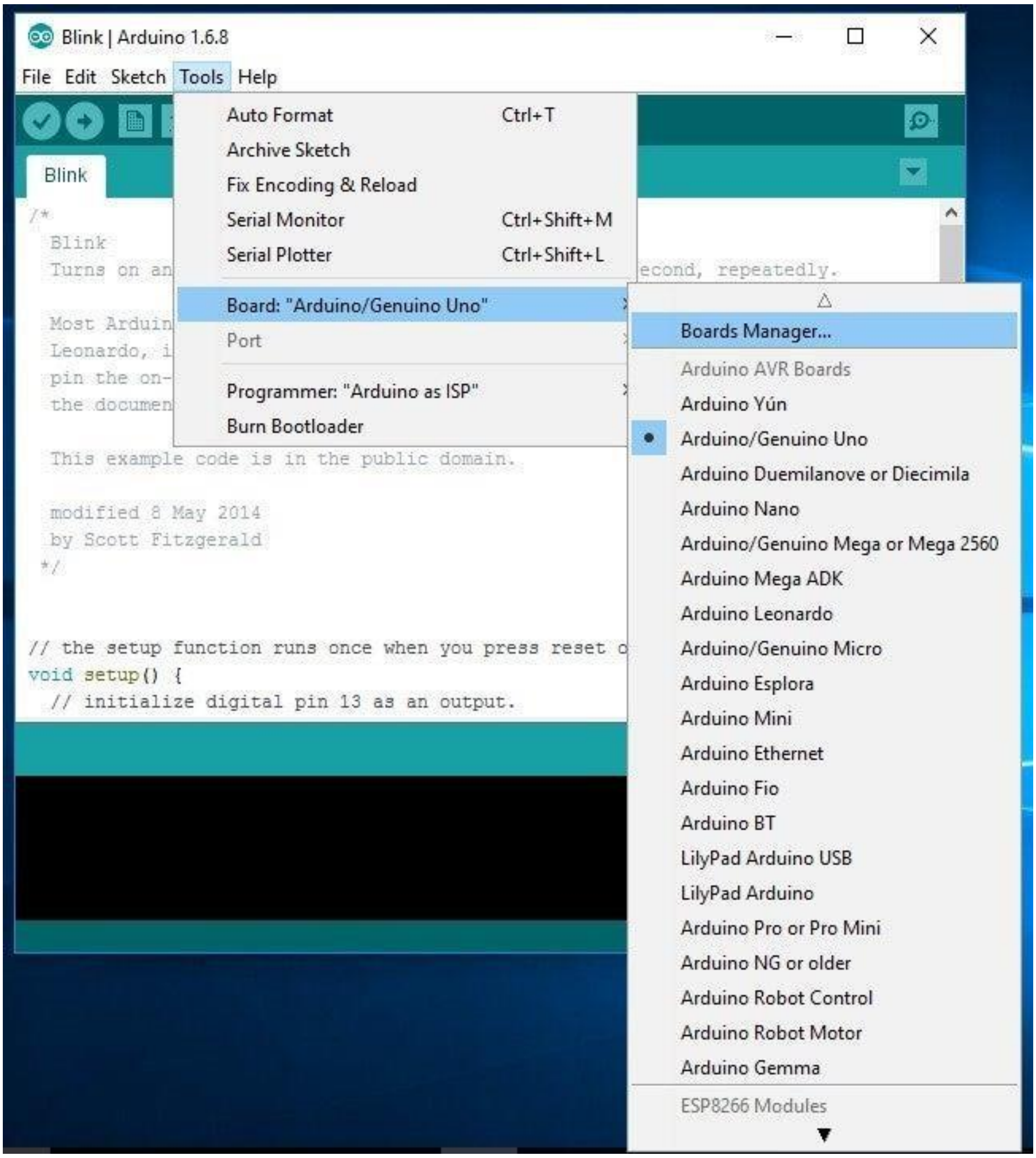

Figure 3.4 Arduino IDE board setup

3.2.4 Arduino IDE COM Port Setup:

In Arduino IDE after plugging in the Arduino board, the USB drivers should have installed automatically. The most recent Arduino IDE should recognize connected boards and label them with which COM port they are using. Select the Tools pulldown menu and then Port. Here it should list all open COM ports, and if there is a recognized Arduino Board, it will also give it's name. Select the Arduino board that you have connected to the PC. If the setup was successful, in the bottom right of the Arduino IDE, you should see the board type and COM number of the board you plan to program. Note: the Arduino Uno occupies the next available COM port; it will not always be COM3.

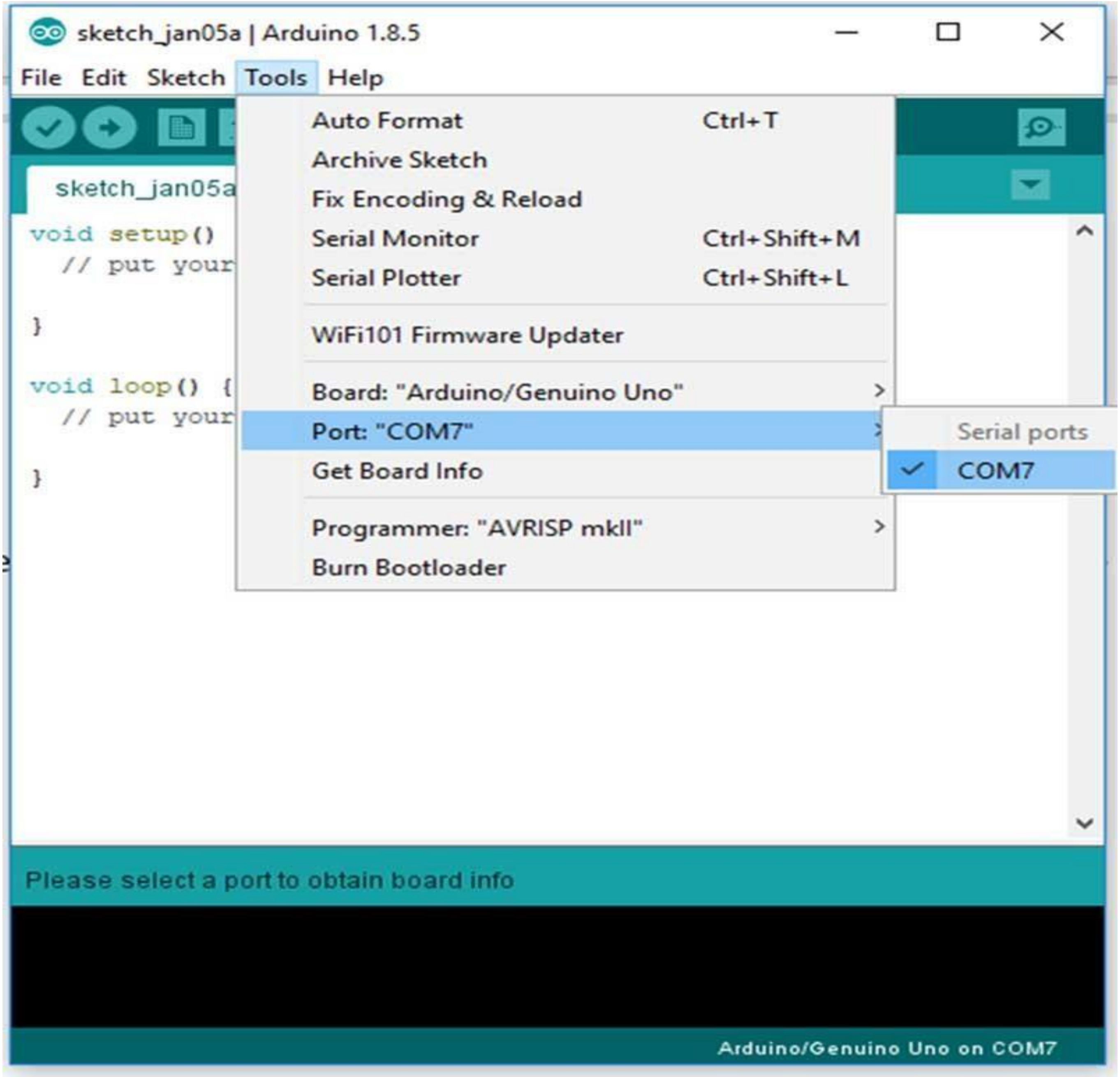

Figure 3.5 Arduino IDE COM Port Setup

3.2.5 Testing the Settings:

One common procedure to test whether the board we are using is properly set up is to upload the "Blink" sketch. This sketch is included with all Arduino IDE releases and can be accessed by the **File** pull-down menu and going to Examples, 01. Basics, and then select Blink. Standard Arduino Boards include a surface-mounted LED labeled "L" or "LED" next to the "RX" and "TX" LEDs, that is connected to digital pin 13. This sketch will blink the LED at a regular interval, and is an easy way to confirm if your board is set up properly and you were successful in uploading code. Open the "Blink" sketch.

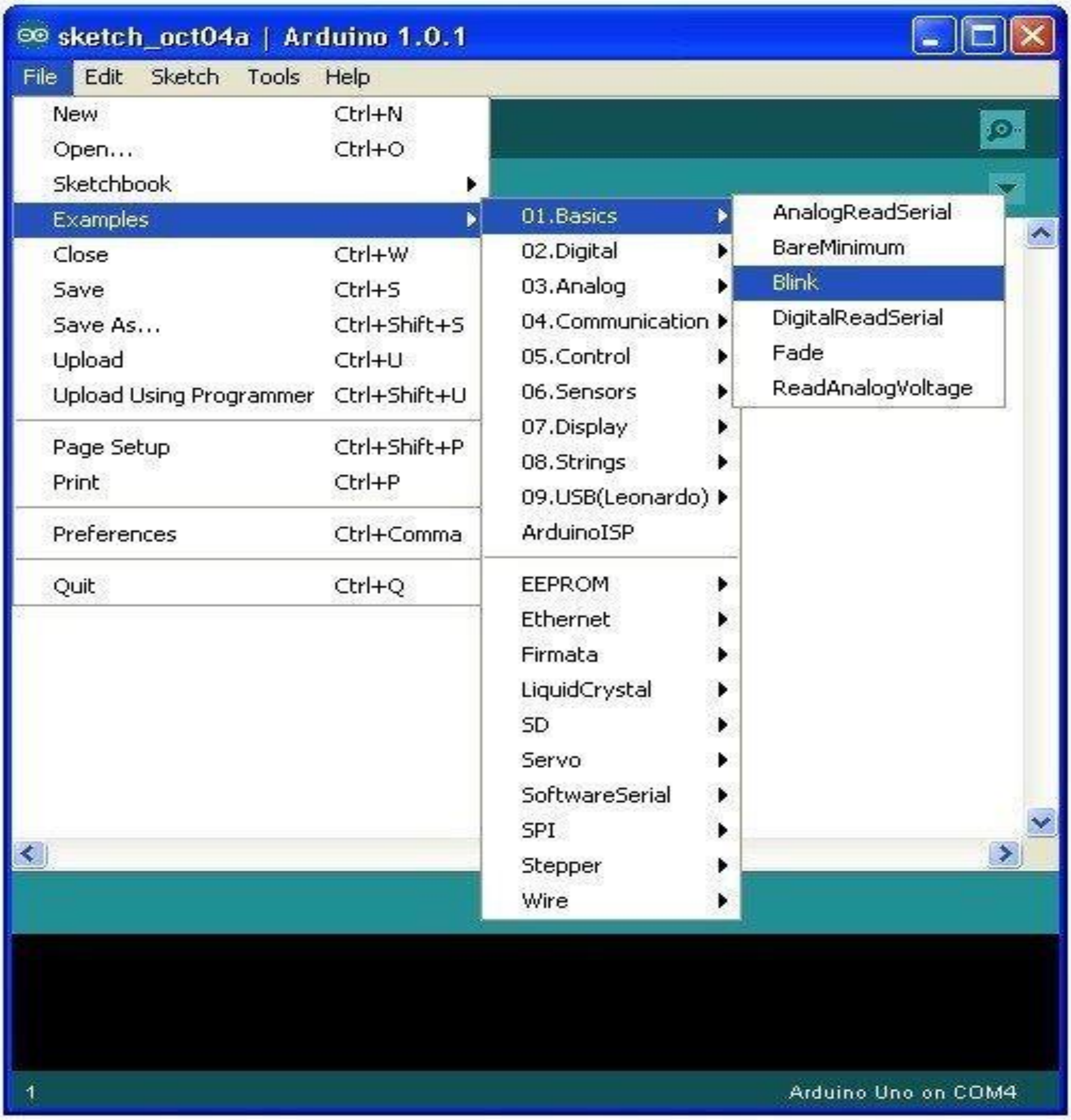

Figure 3.6 Arduino IDE Loading Blink Sketch

After opening the Blink sketch press the "Upload" button in the upper-left corner to upload "Blink" to the board.

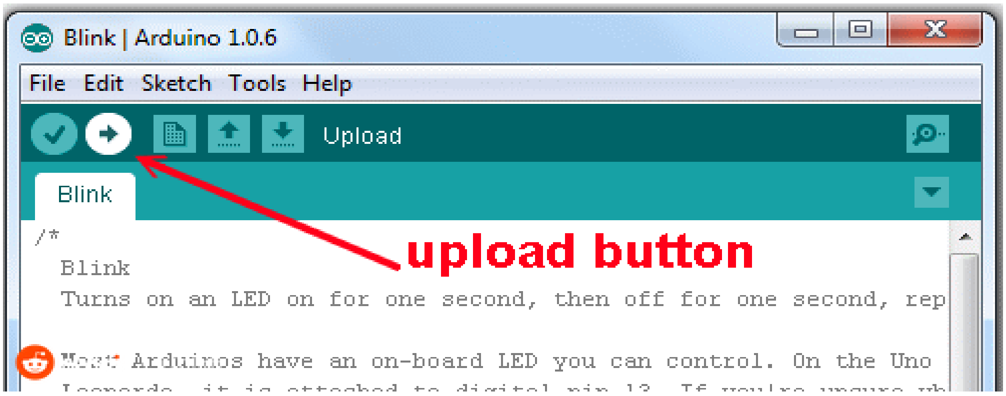

Figure 3.7 Upload button

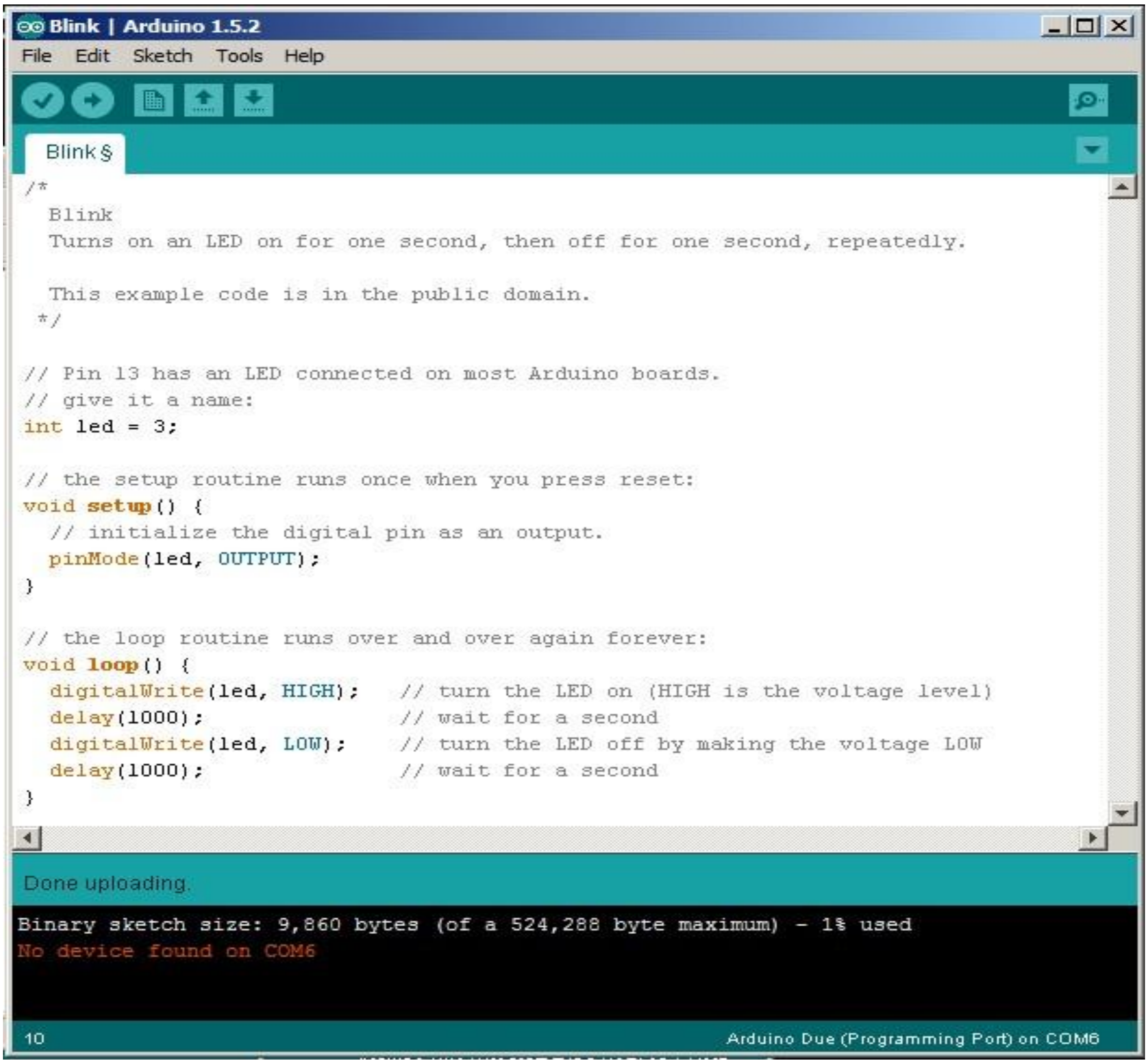

Figure 3.8 Uploading Blink Sketch

3.3 Bluetooth module HC-05:

The **HC-05** is a very good module which can add two-way i.e; full-duplex, wireless functionality to the projects. We can use this module to communicate between two microcontrollers like Arduino or communicate with any device with Bluetooth functionality like a Phone or Laptop. There are many android applications that are already available which makes this process a lot easier. The module communicates with the help of USART at 9600 baud rate hence it is easy to interface with any microcontroller that supports USART. We can also configure the default values of the module by using the command mode. The HC-05 has two operating modes, one is the Data mode in which it can send and receive data from other Bluetooth devices and the other is the AT Command mode where the default device settings can be changed. We can operate the device in either of these two modes by using the key pin as explained in the pin description.

It is very easy to pair the HC-05 module with microcontrollers because it operates using the Serial Port Protocol (SPP). Simply power the module with +5V and connect the Rx pin of the module to the Tx of MCU and Tx pin of module to Rx of MCU.

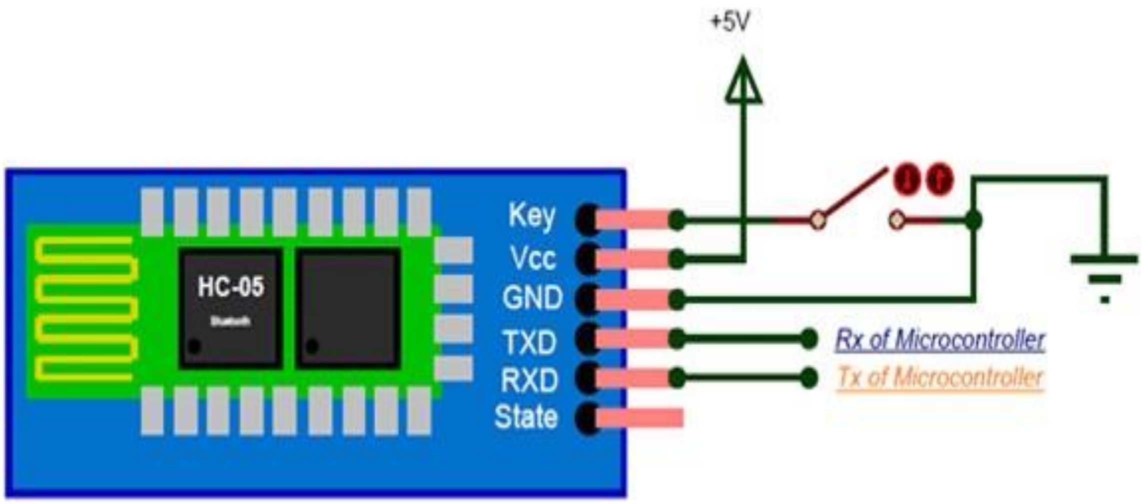

Figure 3.9 HC-05 Bluetooth Module

During power up the key pin can be grounded to enter into Command mode, if left free it will by default enter into the data mode. As soon as the module is powered we should be able to discover the Bluetooth device as “HC-05” then connect with it using the default password 1234 and start communicating with it. The name password and other default parameters can be changed.

3.3.1 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

3.3.2 HC-05 Technical Specifications:

3.3.2.1 Serial Bluetooth module for Arduino and other microcontrollers

3.3.2.2 Operating Voltage: 4V to 6V (Typically +5V)

3.3.2.3 Operating Current: 30mA

3.3.2.4 Range: <100m

3.3.2.5 Works with Serial communication (USART) and TTL compatible

3.3.2.6 Follows IEEE 802.15.1 standardized protocol

3.3.2.7 Uses Frequency-Hopping Spread spectrum (FHSS)

3.3.2.8 Can operate in Master, Slave or Master/Slave mode

3.3.2.9 Can be easily interfaced with Laptop or Mobile phones with Bluetooth

3.3.2.10 Supported baud rate: 9600,19200,38400,57600,115200,230400,460800.

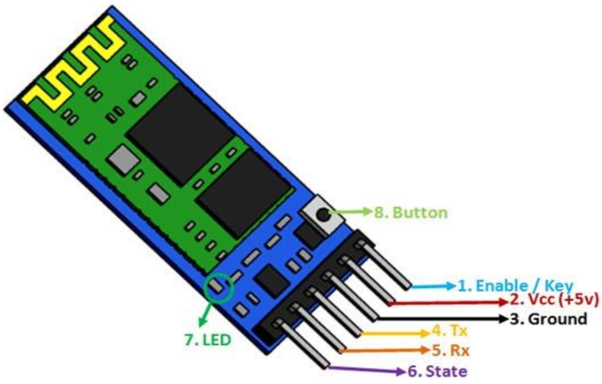

Figure 3.10 HC05 Pin Configuration

This module is cheap, readily available, and quick to acquire. It is also designed to ensure compatibility with Arduino microcontrollers. Furthermore, it is highly customizable, such as allowing its name to be changed, modifying its access password, and switching from master mode to slave mode. This last feature is important because it will enable future development of the Sensor Interface to add wireless Bluetooth sensor reading functionality. To communicate with Bluetooth sensors, the module should be in Master mode. However, when communicating with the Android device, the module should be in Slave mode. This Bluetooth module's ability to switch between these 2 modes will allow the Sensor Interface to communicate with Bluetooth sensors as wired sensors.

The Bluetooth module's RX pin must connect with the microcontroller's TX pin, and its TX pin must connect to the microcontroller's RX pin. The Arduino Uno's default TX serial pin is P1 and the default RX pin is P0. However, these were not used for a couple of reasons. Most importantly, when P1 and P0 were used as TX and RX pins, the Bluetooth module would not communicate with the microcontroller. This meant that different pins had to be used. Fortunately, an Arduino library exists to change digital pins to RX/TX pins, called SoftwareSerial. Using this library, P3 was changed to a TX pin and P2 was changed to an RX pin. This decision has the added benefit of freeing up P1 and P0. This is beneficial because the Arduino has a useful debugging and monitoring feature called Serial Monitor. Using this, the Arduino can write messages to a computer screen when connected to a computer.

However, this feature only works when P1 and P0 are not in use. Therefore, by using P3 and P2 for TX and RX instead of P1 and P0, testing and debugging was a much easier and faster process. The Bluetooth module has one more input pin, labeled KEY. This pin is used for changing its mode between Master mode and Slave mode, as well as customizing its other 19 properties. These options are not used currently, but they may be in the future. Therefore, this pin is connected to P4 on the Arduino Uno if property changes are ever desired.

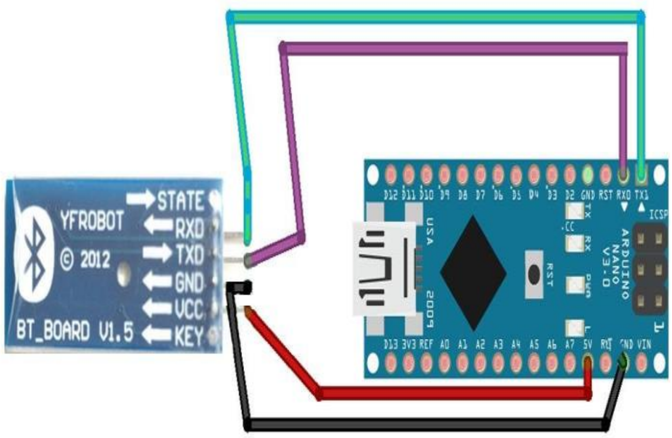

Figure 3.11 Microcontroller - Bluetooth Module Connectivity

3.3.3 Pin Configuration:

Pin No	Pin Name	Description
1	Enable / Key	This pin is used to toggle between Data Mode (set low) and AT command mode (set high). By default it is in Data mode
2	Vcc	Powers the module. Connect to +5V Supply voltage
3	Ground	Ground pin of module, connect to system ground.
4	TX – Transmitter	Transmits Serial Data. Everything received via Bluetooth will be given out by this pin as serial data.
5	RX – Receiver	Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth
6	State	The state pin is connected to on board LED, it can be used as a feedback to check if Bluetooth is working properly.
7	LED	Indicates the status of Module are:- • Blink once in 2 sec: Module has entered Command Mode • Repeated Blinking: Waiting for connection in Data Mode • Blink twice in 1 sec: Connection successful in Data Mode
8	Button	Used to control the Key/Enable pin to toggle between Data and command Mode

Table 3.1 Bluetooth Pin Description

3.4 Robot Chassis:

A chassis is the load-bearing framework of an artificial object, which structurally supports the object in its construction and function. An example of a chassis is a vehicle frame, the underpart of a motor vehicle, on which the body is mounted; if the running gear such as wheels and transmission, and sometimes even the driver's seat, are included, then the assembly is described as a rolling chassis.

Figure 3.12 Robot Chassis

3.5 DC Motors:

A DC motor is a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor.

DC motors were the first form of motor widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight brushed motor used for portable power tools and appliances. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills.

The advent of power electronics has made replacement of DC motors with AC motors possible in many applications.

Figure 3.13 DC Motor

3.6 Wheels:

Wheels are used in movement and change of direction by varying the relative rate of rotation. To balance the robot, additional wheels or casters may be added. If both the wheels are driven in the same direction and speed, the robot will go in a straight line.

Figure 3.14 Wheels

3.7 L293D IC:

L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors. It contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state.

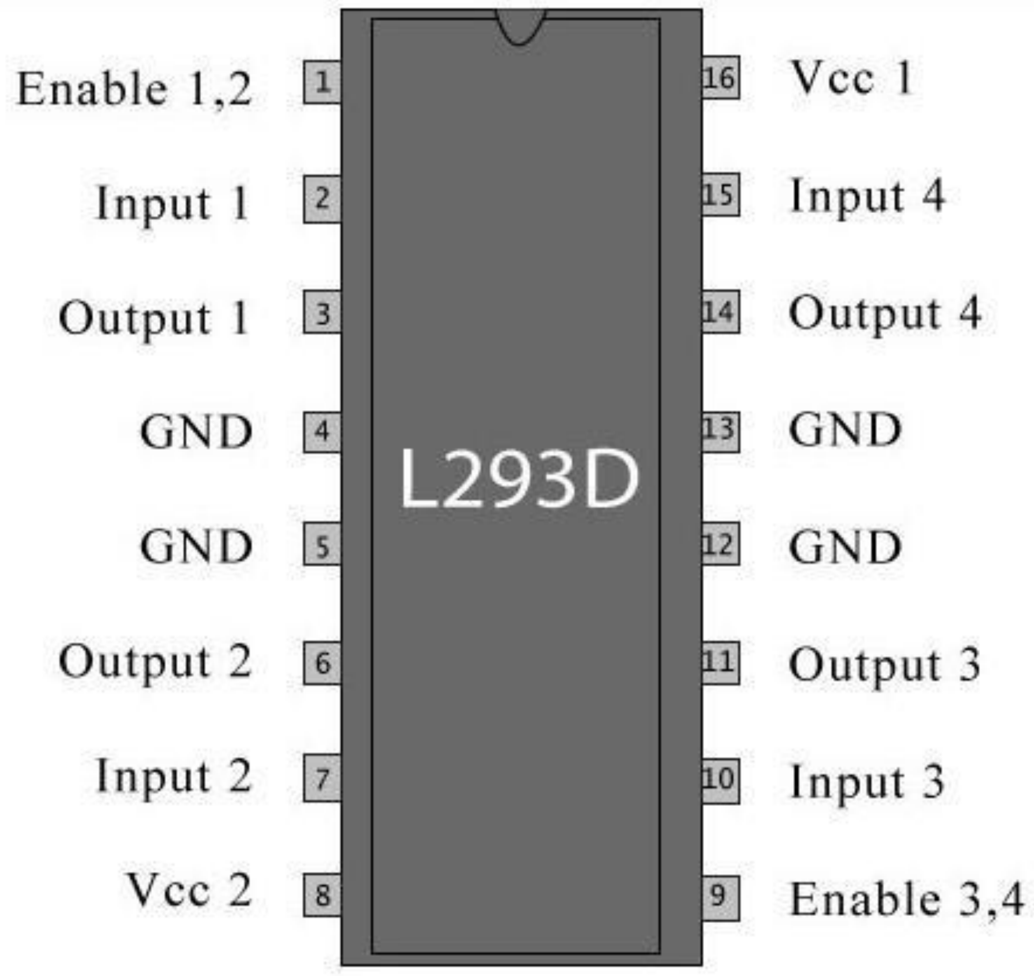

Figure 3.15 L293D IC Pin diagram

3.7.1 Pin Description:

Pin No	Function	Name
1	Enable pin for Motor 1; active high	Enable 1,2
2	Input 1 for Motor 1	Input 1
3	Output 1 for Motor 1	Output 1
4	Ground (0V)	Ground
5	Ground (0V)	Ground
6	Output 2 for Motor 1	Output 2
7	Input 2 for Motor 1	Input 2
8	Supply voltage for Motors; 9-12V (up to 36V)	Vcc 2
9	Enable pin for Motor 2; active high	Enable 3,4
10	Input 1 for Motor 1	Input 3
11	Output 1 for Motor 1	Output 3
12	Ground (0V)	Ground
13	Ground (0V)	Ground
14	Output 2 for Motor 1	Output 4
15	Input2 for Motor 1	Input 4
16	Supply voltage; 5V (up to 36V)	Vcc 1

Table 3.2 L293D Pin description

3.8 Bread board:

A breadboard is a simple device designed to let you create circuits without the need for soldering. They come in various sizes, and the design can vary.

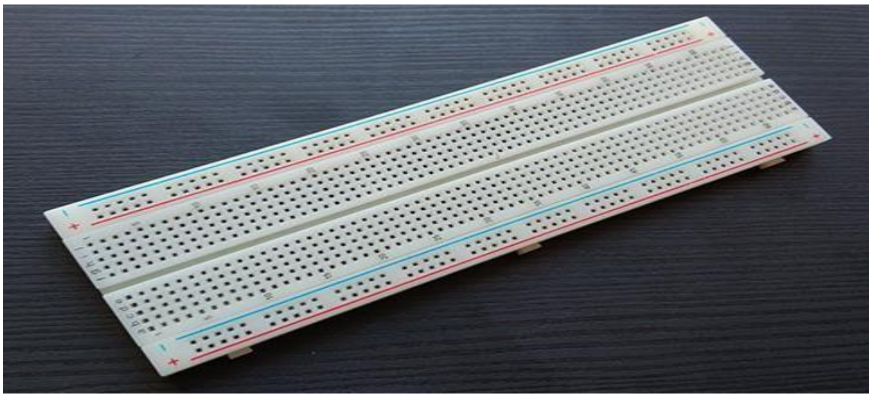

Figure 3.16 Bread Board

The power rails run horizontally as two rows at the top and bottom. Meanwhile, the vertical columns run inwards as you move down the board.

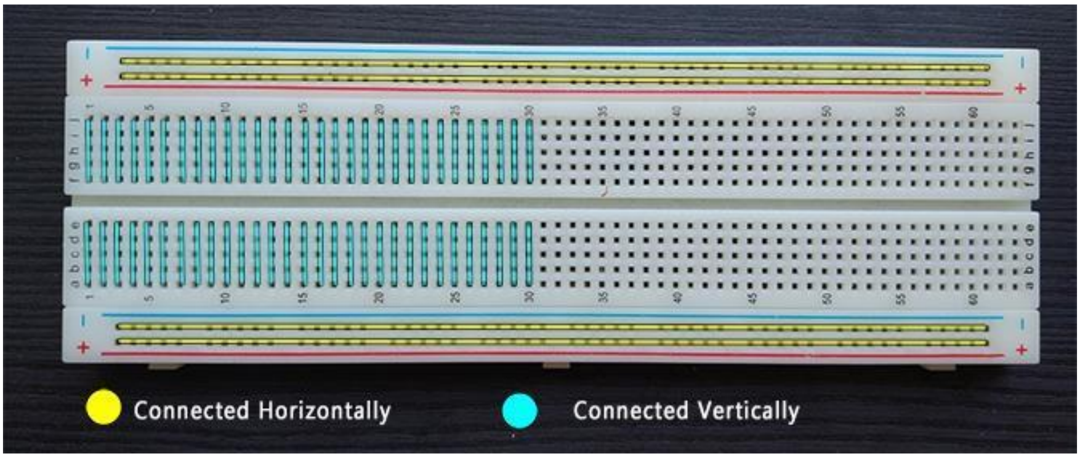

Figure 3.17 Horizontal and Vertical connections in Bread board

3.9 Battery 12 volts:

A rechargeable 12 volt battery is used in the prototype and has characteristics of high discharge rate, wide operating temperature, long service life and deep discharge recover.

Figure 3.18 12 volt battery

3.10 Voltage Regulator-7805:

A voltage regulator IC maintains the output voltage at a constant value. 7805 Voltage Regulator, a member of 78xx series of fixed linear voltage regulators used to maintain such fluctuations, is a popular voltage regulator integrated circuit (IC). The xx in 78xx indicates the output voltage it provides. 7805 IC provides +5 volts regulated power supply with provisions to add heat sink.

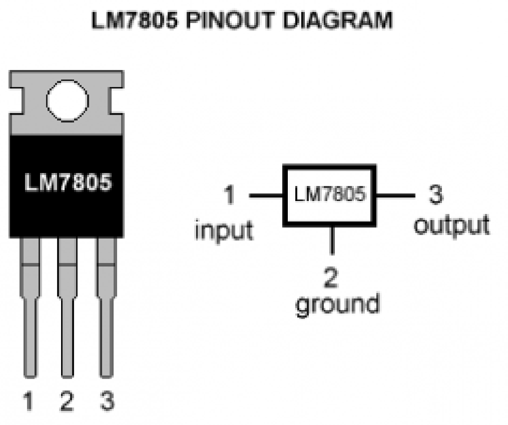

Figure 3.19 IC 7805

3.10.1 PIN DESCRIPTION:

Pin Number	Pin	Function	Description
1	Input	Input voltage (7V-35V)	In this pin of the IC positive unregulated voltage is given in regulation.
2	Ground	Ground (0V)	In this pin where the ground is given. This pin is neutral for equally the input and output.
3	Output	Regulated output; 5V (4.8V-5.2V)	The output of the regulated 5V volt is taken out at this pin of the IC regulator.

Table 3.3 7805 Pin description

3.11 Jumpers:

A jump wire also known as jumper wire, or jumper is an electrical wire, or group of them in a cable, with a connector or pin at each end, which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering. Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.

3.11.1Types:

There are different types of jumper wires. Some have the same type of electrical connector at both ends, while others have different connectors. Some common connectors are:

- **Solid tips**: are used to connect on/with a breadboard or female header connector. The arrangement of the elements and ease of insertion on a breadboard allows increasing the mounting density of both components and jump wires without fear of short-circuits. The jump wires vary in size and colour to distinguish the different working signals.
- **Crocodile clips** – are used, among other applications, to temporarily bridge sensors, buttons and other elements of prototypes with components or equipment that have arbitrary connectors, wires, screw terminals, etc.
- **Banana connectors** – are commonly used on test equipment for DC and low-frequency AC signals.
- **RCA connectors** – are often used for audio, low-resolution composite video signals, or other low-frequency applications requiring a shielded cable.
- **RF connectors** – are used to carry radio frequency signals between circuits, test equipment, and antennas.

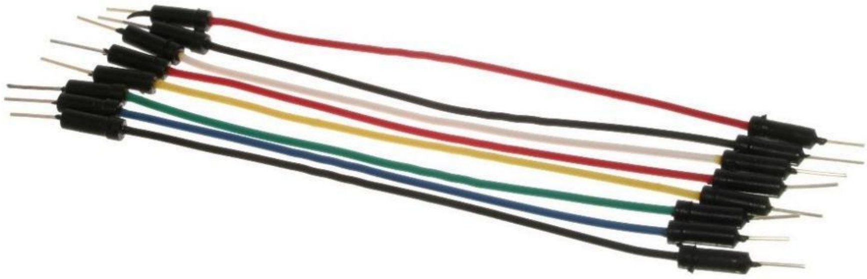

Figure 3.20 Jumper Wire

CHAPTER 4

WORKING

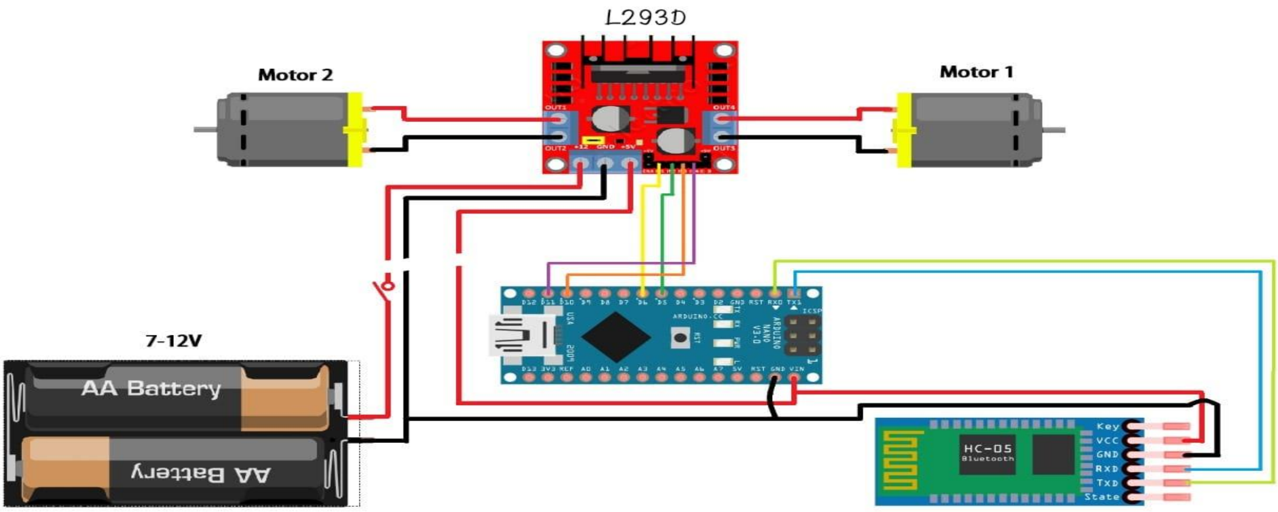

Fig 4.1 Circuit Diagram

Gesture recognition is the main aim of this work. The user holds an Android operated Smartphone, and moves, or rotates his hand in any direction. The accelerometer within the phone is regulated to generate a maximum and minimum value for the movement of the hand in three dimensional co- ordinates depending upon the external environmental conditions. The android application does the work of sensing the accelerometer calibration and generating the maximum and minimum values from it. Depending upon the values obtained, it sends a determinant value to the microcontroller using Bluetooth. The Bluetooth module receives data and transmits it to Arduino where it checks the determinant value and moves the robot accordingly. The whole process is under an infinite loop, so it runs as long as the power is supplied. The output depends on the accelerometer inputs directly that can be used to control the robot. The accelerometer input depends on the gestures of the user's hand.

The steps stated above are broadly described in this section. The system consists of the following steps to work as mentioned:

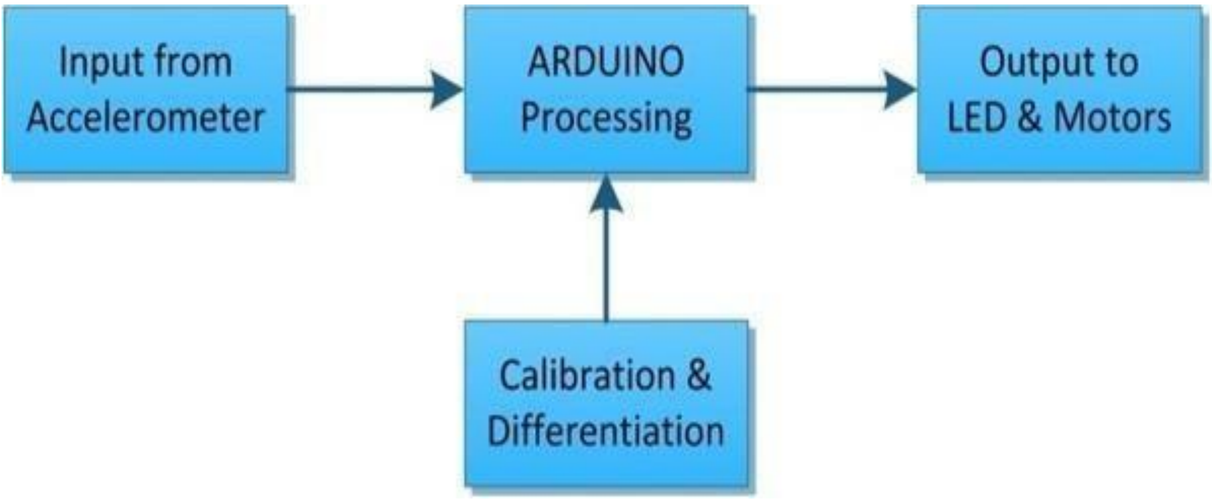

Figure 4.2: Block diagram of the system

The input to the application is the direction of movement of hand of the user given by the accelerometer. This is analog in nature. It is then digitally coded by the Android application before sending it to Arduino by HC-05 Bluetooth module. The signal goes to the digital pins of the Arduino board, which has an inbuilt AD/DA converter of 8 bit. The Arduino process the received data. Based on the data received, appropriate signal is transmitted to the motor driver to rotate the motor in such a way that the robot moves in the direction of movement

4.1 Movement of Motors and Wheels:

There are four DC motors used in the design of this robot: one motor for each wheel. The motors are controlled by the Adafruit motor shield. The shield is stacked on top of Arduino. Every shield stacked can run 4 DC motors. Installing the Adafruit Motor Shield library gives the flexibility of using the motors just by calling some pre-defined functions as motor1, set speed that sets the speed of the motor to 250 rpm, or motor1, run(FORWARD) that makes the motor1 to rotate forward. These functions are called from the program burnt in the Arduino microcontroller. The signal is sent to the motor shield that runs the motors.

The wheels are connected to the motors. 2 DC motors are used one for left wheels, and one for right wheels. When the signal received in the motor shield is to move forward, all the wheels of motors rotate forward, this turns all the four wheels in the forward direction. The robot moves in the forward direction. When the signal received in the motor shield is to turn the robot in the forward left direction, the left diagonal motors are rotated backwards while the right diagonal motors are made rotated forwards. This makes the robot turn in the forward left direction. When the signal received in the motor shield is to turn the robot in the forward right direction, the right diagonal motors are rotated backward while the left diagonal motors are rotated forwards. This makes the robot turn in the forward right direction. When the signal in the motor shield is to move backward, both the pairs of the motors are rotated backwards resulting the robot to move backwards. When the signal in the motor shield is to stop the robot, all the motors are made stationary resulting the robot to stop. Similarly, to rotate the robot in backward directions, similar methodology is used. To turn the robot in the backward left direction, the left diagonal motors are rotated forwards while the right diagonal motors are rotated backwards. This makes the robot turn in the backward left direction. To turn the robot in the backward right direction, the right diagonal motors are rotated forwards while the left diagonal motors are rotated backwards. This makes the robot turn in the backward right direction…..

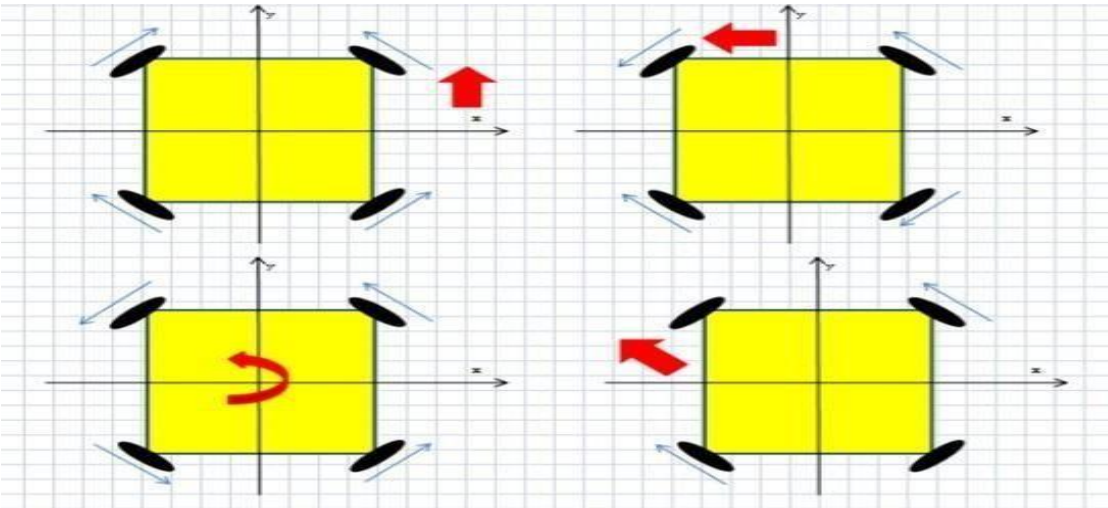

Figure 4.3 Movement of the Motors and Wheels

4.2 Android Application:

The Android application is the key to control the robot using hand gestures. The application reads the accelerometer state and X, Y, and Z values are obtained in the application. There are two threshold values assigned for each movement: one is the MAX_THRESHOLD, and the other is the MIN_THRESHOLD. If the obtained value lies between these thresholds of a certain movement, then the character assigned to denote that movement, which is called the DET or determinant is sent to the robot via Bluetooth. The application continuously sense this until the application is ON. A graphical user interface has been designed for the comfort of the user. The application abstracts the calculations and accelerometer values, but the user interface shows the direction of movement of the hand so that the user is aware of wrong turns in the bot.

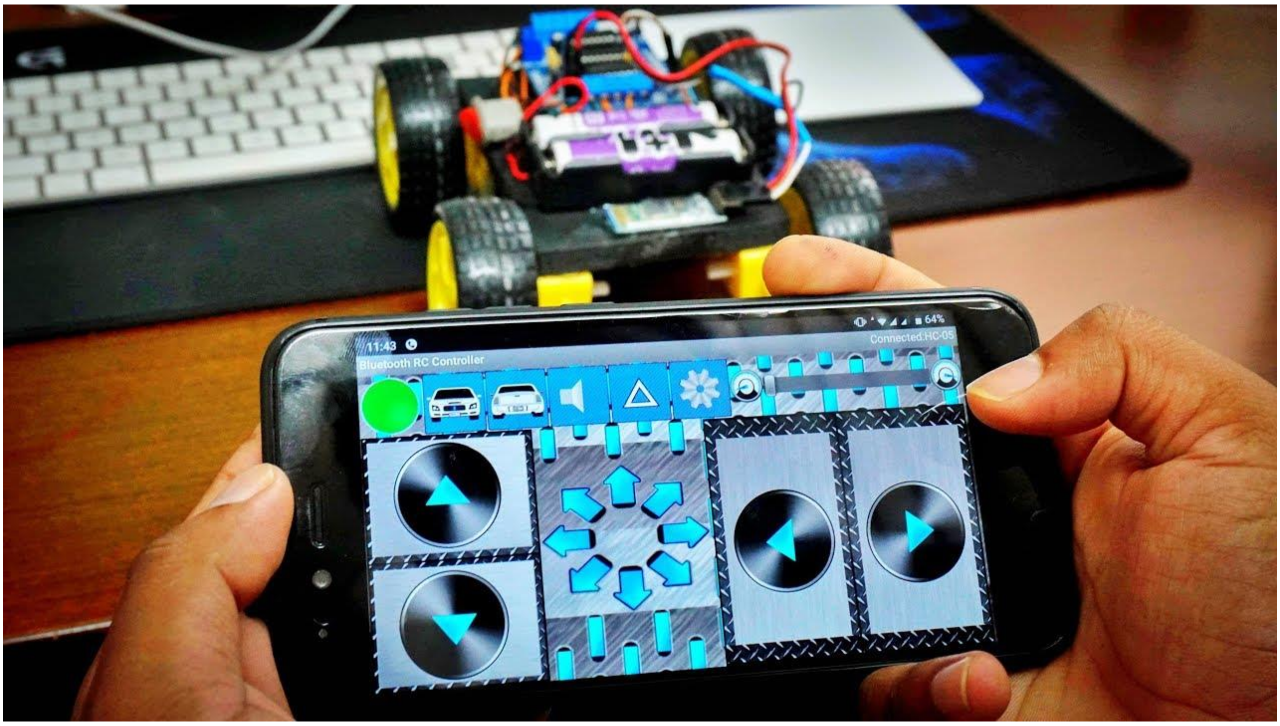

Figure 4.4 Screenshot of the Android application

CHAPTER 5

SOURCE CODE

```
//Motor A
const int motorPin1F  = 3;  // Pin 2 of L293
const int motorPin1B  = 4;  // Pin 7 of L293

//Motor B
const int motorPin2F  = 5; // Pin  10 of L293
const int motorPin2B  = 6;  // Pin  15 of L293

//buzzer
const int Buzzer  = 12;

void setup()
{
   Serial.begin(9600); // baud rate

   pinMode(Buzzer, OUTPUT);

   pinMode(motorPin1F, OUTPUT);
   pinMode(motorPin1B, OUTPUT);
   pinMode(motorPin2F, OUTPUT);
   pinMode(motorPin2B, OUTPUT);
```

```
    digitalWrite(motorPin1F, LOW);
    digitalWrite(motorPin1B, LOW);
    digitalWrite(motorPin2F, LOW);
    digitalWrite(motorPin2B, LOW);
}

void loop()
{
  while (Serial.available())
  {
    int a = Serial.read();

     switch (a)
     {

       case 'F':
       digitalWrite(motorPin1F, HIGH);
       digitalWrite(motorPin2F, HIGH);
       break;

       case 'B':
       digitalWrite(motorPin1B, HIGH);
       digitalWrite(motorPin2B, HIGH);
       break;
```

```
case 'L':
digitalWrite(motorPin1F, HIGH);
break;

case 'R':
digitalWrite(motorPin2F, HIGH);
break;


case 'V':
digitalWrite(Buzzer, HIGH);
break;

case 'v':
digitalWrite(Buzzer, LOW);
break;

default:
digitalWrite(motorPin1F, LOW);
digitalWrite(motorPin1B, LOW);
digitalWrite(motorPin2F, LOW);
digitalWrite(motorPin2B, LOW);
break;
```

```
        }
      }
    }
```

CHAPTER 6

APPLICATIONS

- Used to control wheel chair.
- Used for home appliances.
- Used as a robot assistance.
- Indoor positioning systems.
- Lawn/gardening management.
- Medical applications for the purpose of surgery.
- In industries to control trolley and lift.
- Used in the construction field.

CHAPTER 7

CONCLUSION

The design and implementation of Accelerometer Controlled Robot is presented and developed using Arduino microcontroller and Android Smartphone. The built device is cheap, and is easy to carry from one place to another. The addition of the some additional sensors or camera will make it more productive. The limitation of the hardware being associated with a system has been reduced to a great extentt. The sensors are intended to replace the remote control that is generally used to run the car.It will allow us to control the forward and backward, and left and right movements, while using the same accelerometer sensor to control the throttle of the car based on the accelerometer movements.

CHAPTER 8

FUTURE SCOPE

The accelerometer controlled robot designed can be used for surveillance purpose. The robot can be applied in a wheelchair where the wheelchair can be driven by the movements of rider's hand. Wi-Fi can be used for communication instead of Bluetooth to access it from a greater distance. Edge sensors can be incorporated to it to prevent the robot from falling from any surface. Some camera can be installed which can record and send data to the nearby computer or cell-phone. It can be implemented on a watch, or in any home appliances like Room heater. Modern ARDUINO chips support Intranet as well as Internet connections which can be utilized to a greater extent. This robotic car can be enhanced to work in the military surveillance where it can be sent to some enemy camps and track it's activities via Internet.

CHAPTER 9

REFERENCES

1. http://www.robotplatform.com/howto/L293/motor_driver_1.html
2. https://circuitdigest.com/microcontroller-projects/rf-remote-controlled-leds-usingraspberry-pi
3. https://www.engineersgarage.com/sites/default/files/HT12D_1.jpg
4. https://www.engineersgarage.com/sites/default/files/HT12E.jpg
5. https://docs.particle.io/assets/images/pi-pinout-diagram-01.png
6. https://www.dhresource.com/260x260s/f2-albu-g5-M00-50-59-rBVaJFlHom2AH1_1AAQOVIqxkAE259.jpg/6v-bend-of-uniaxial-gear-motor-ttdeceleration.jpg