A MINI PROJECT ON

"Android Phone Controlled Arduino Based Robot Using Bluetooth"

Submitted in partial fulfillment of requirement in

"MECHANICAL MEASUREMENTS AND CONTROLS"

Submitted By:

1. DARSHAN HEGDE (131)

2. SUSHIL HEDAU (130)

3. SAURABH HYALINGE (133)

4. ADWAIT ATHAVALE (103)

5. AKSHAY DORLE (122)

THIRD YEAR ENGINEERING IN MECHANICAL (SEM V)

GUIDED BY

Prof. NITIN NANDESHWAR

YADAVRAO TASGAONKAR COLLEGE OF ENGINEERING AND MANAGEMENT

DEPARTMENT OF MECHANICAL ENGINEERING

Academic Year 2016-2017

CERTIFICATE

This is to certify that this report entitled

"Android Phone Controlled Arduino Based Robot Using Bluetooth"

Submitted by the following students of

"Mechanical Engineering"

Towards the partial fulfillment in the requirements in

"MECHANICAL MEASUREMENTS AND CONTROLS"

1. **DARSHAN HEGDE (131)**
2. **SUSHIL HEDAU (130)**
3. **SAURABH HYALINGE (133)**
4. **ADWAIT ATHAVALE (103)**
5. **AKSHAY DORLE (122)**

Throughout their tenure of completion of task they have been guided and assessed by me, I am satisfied that their contribution was proportionate, they were satisfactory progressive and their task is up to standard envisaged by University of Mumbai.

DATE

PROJECT GUIDE

(Prof. NITIN NANDESHWAR)

HOD
(Prof. R.K.AGRAWAL)

PRINCIPAL
(Dr. A. K. Sen)

TABLE OF CONTENT

ACKNOWLEDGEMENT

"We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and web research. We would like to extend my sincere thanks to all of them.

We are highly indebted to **Prof. Nitin Nandeshwar** for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

We would like to express our gratitude towards my parents & member of **YADAVRAO TASGONKAR COLLEGE OF ENGINEERING AND MANAGEMENT** for their kind co-operation and encouragement which help me in completion of this project.

Ours thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities."

INTRODUCTION

Arduino is composed of two major parts: the Arduino board, which is the piece of hardware you work on when you build your objects; andthe Arduino IDE, the piece of software you run on your computer. You use the IDE to create a sketch (a little computer program) that you upload to the Arduino board.

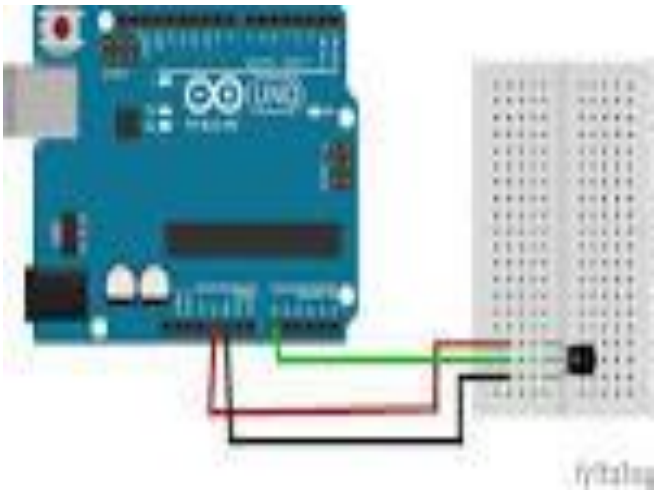


Figure 05: Arduino Development Module

Fig(1)

The first Arduino was introduced in 2005, aiming to provide an inexpensive and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors.

Hardware An official Arduino Uno with descriptions of the I/O locations An early Arduino board[8] with anRS-232 serial interface (upper left)and an Atmel ATmega8 microcontrollerchip (black, lower right); the 14 digitalI/O pins are located at the top and thesix analog input pins at the lower right.An Arduino board consists of an Atmel 8-, 16- or 32-bit AVR microcontroller withcomplementary components that facilitateprogramming and incorporation into othercircuits. An important aspect of the Arduino is its standard connectors, which lets usersconnect the CPU board to a variety ofinterchangeable add-on modules known as shields,Some shields communicate with the Arduino board directly over various pins, butmany shields are individually addressable viaan I^2C serial bus—so many shields can bestacked and used in parallel.

Official Arduino shave used the mega AVR series of chips, specifically the Atmega8 , Atmega168 ,Atmega328 , Atmega1280 , and Atmega2560 .A handful of other processors have been used by Arduino compatibles. Most boards include a5 volt linear regulator and a 16 MHz crystaloscillator (or ceramic resonator in somevariants), although some designs such as theLilyPad run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

An Arduino's microcontroller is also pre-programmed with aboot loader that simplifies uploading ofprograms to the on-chip flash memory,compared with other devices that typicallyneed an external programmer . This makes using an Arduino more straightforward by allowing the use of an ordinary computer as the programmer. Currently, opti boot loader is the default boot loader installed on Arduino UNO.

Abstract:

It is through efficient electronic programming that a computer can control a robot, hence a robot can be thought of as an Electromechanical machine. Some of the essential characteristics that a robot must have are – sensing, movement, energy, intelligence. It performs a task using control systems, various power supplies and software all working together. We developed an Android application which uses remote buttons to guide an RC car's motion. Hence, the mobile device harbouring the Android application acts as the car's remote control. Bluetooth is the basis of communication between the controller and Android, using the USART protocol.

Further Introduction:

The very primitive concept of smartphones is believed to have been envisioned back in the mid- 1970s, but that vision didn't come into fruition until 1992 when IBM first showed its face. Nowadays they come with richer entertainment Function, efficient communication methods and reinforced Processors. As Bluetooth is used for Data Exchange, it is through this data exchange capability that through Bluetooth, Devices are now being controlled and monitored. Bluetooth technology was created in 1994 by the telecom dealer "Ericsson" for integrating with Smart phones. But through the years, with dramatic increase in smart phone Users, 8luetooth has turned them into all-purpose portable devices by redefining the world of data exchange and transferring wired devices into wireless devices; capable of efficient communication and the fact that host Bluetooth device is capable of communicating with as many as seven Bluetooth modules simultaneously through one link is proof enough. Android is wide spread and influential in today's scenario, that using a smart phone as the 'brain' of a robot is already an active research field, providing a number of opportunities and possibilities.

We've projected this research work to provide simpler hardware architecture, but with powerful and concise computational platforms required to build the Robot. Our purpose on educational robotics is simple architecture so as to serve the students an elegant idea so that they can build their own robots at low cost and use them as a decent platform for experiments in several courses, also aid the robot's designer to focus on their research instead of Bluetooth connection infrastructure. The following list shows the typical robot control architecture

Requirements and Design :

The Following parts are required for this project:

1) Arduino Uno R3

2) HC-05

3) 2 DC – Motors

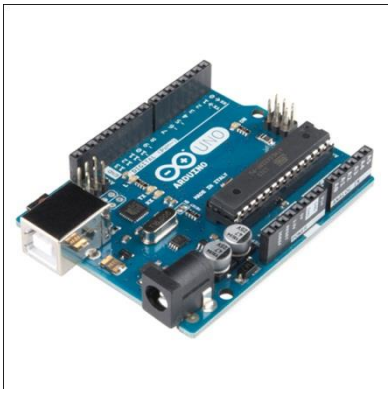
4) Breadboard

5) L293D Motor Driver Board

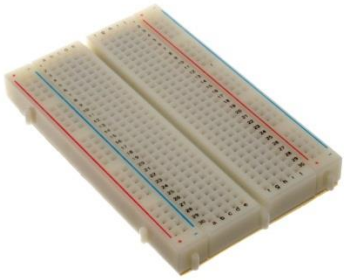
6) 9V Batteries, 4x1.5V Batteries with battery Caps (2 battery and 2 caps)

7) Jumper Cables, Connecting wires

8) Android Phone for controlling the bot



Arduino Uno R3



Breadboard

HC-05 Bluetooth Module

Geared DC-motor

IC-L293D

Fig(2)

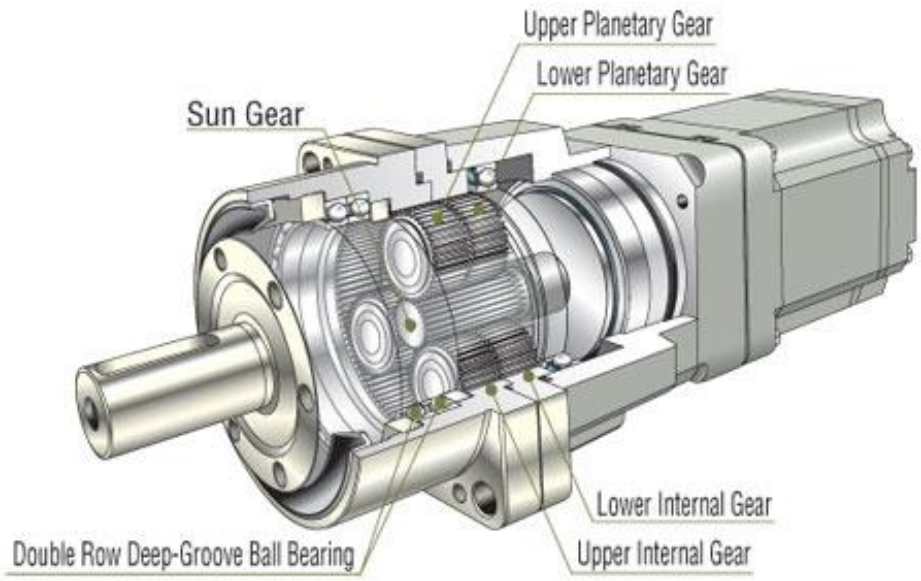
Construction

1. DC Motors:

Electric Motors produce Mechanical movements by taking in the electrical energy and producing equivalent mechanical energy. Hundreds and thousands of devices are powered by electrical motors- from small pick-and-place robots to big turbines- motors find applications in every industry.

For our project we have used geared DC motors .Sometimes the usual DC motors found in normal RC controlled toy cars are insufficient to provide the required torque for the robot.

Therefore geared DC-motors come in handy when getting the required torque for the required RPM of the motors for the robot is needed.



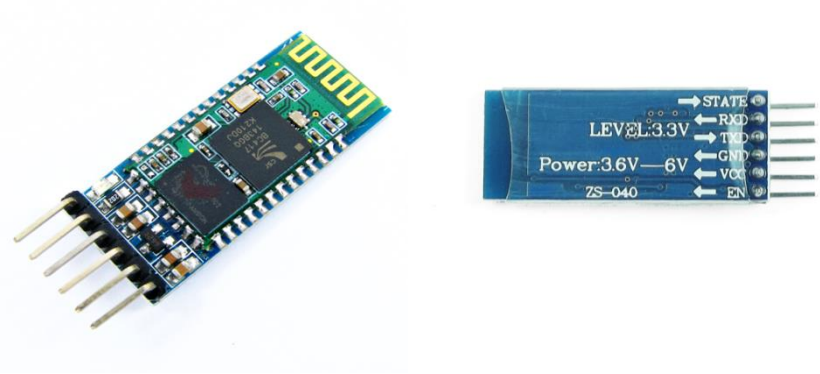
Section View of a geared DC-Motor

Fig(3)

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.

Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.



Fig(4)

Specifications:

Hardware features:

- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- With edge connector

Software features:

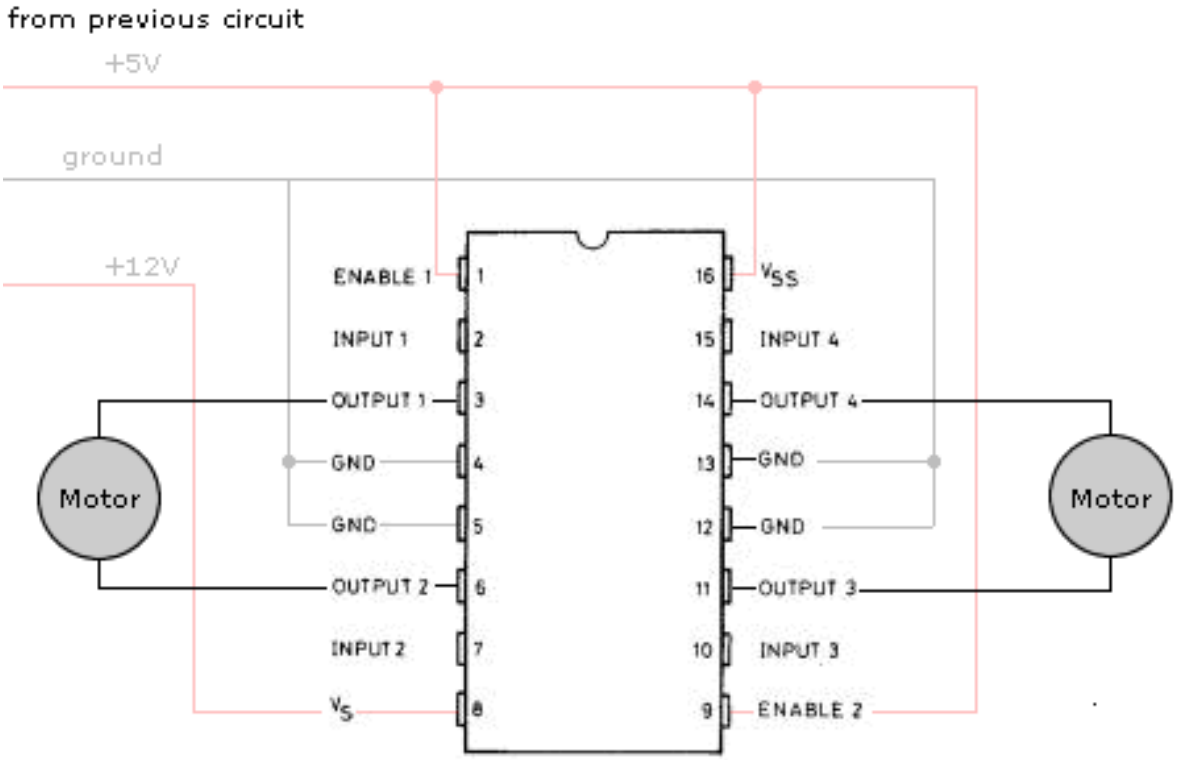
- Default Baud rate: 38400, Data bits:8, Stop bit:1,Parity:No parity, Data control: has. Supported baud rate: 9600,19200,38400,57600,115200,230400,460800.
- Given a rising pulse in PIO0, device will be disconnected.
- Status instruction port PIO1: low-disconnected, high-connected;
- PIO10 and PIO11 can be connected to red and blue led separately. When master and slave are paired, red and blue led blinks 1time/2s in interval, while disconnected only blue led blinks 2times/s.
- Auto-connect to the last device on power as default.
- Permit pairing device to connect as default.
- Auto-pairing PINCODE:”0000” as default
- Auto-reconnect in 30 min when disconnected as a result of beyond the range of connection.

Motor Shield:

DESCRIPTION:

The Device is a monolithic integrated high voltage,high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoides, DC and stepping motors) and switching power transistors.
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.
This device is suitable for use in switching applications at frequencies up to 5 kHz.

The circuit diagram is shown below to connect 2 DC-Motors with the L293D



Fig(5)

By using the Motor Shield, we're basically adding on capabilities on to the Arduino. It is better than using just the Arduino as in that case, a whole lot of more pins are

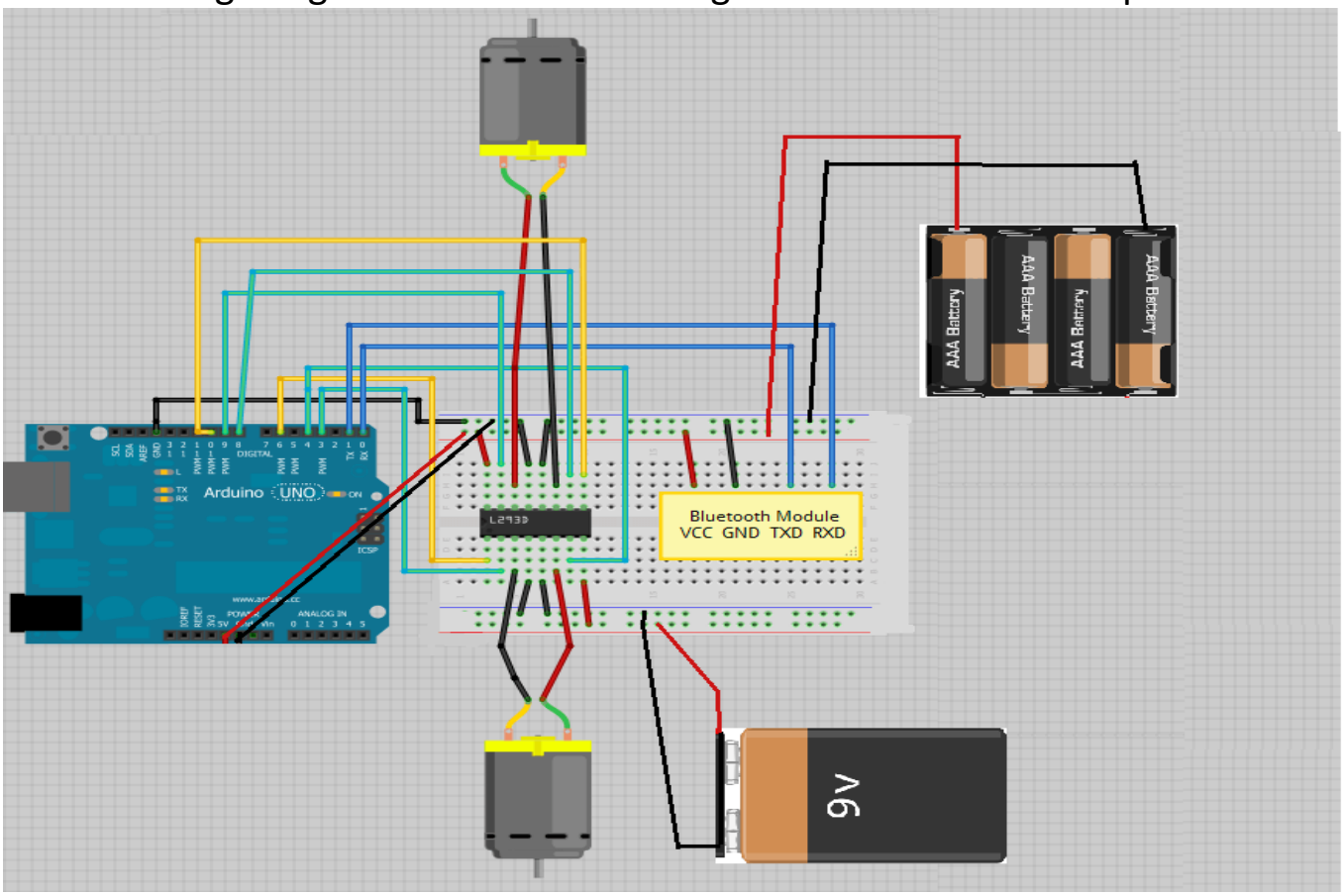
required, not to forget the power issues. The Motor Control chips allow much more precise control over the DC Motors, which include reversing etc. The Motor Shield can supply up to 12 V, 2 ampere per motor channel. Motor Shield pins are used to plug into the Female headers of the Arduino. There are two sets of connectors on either sides of the shield and each of them controls either one stepper motor, or two DC motors.

Other components:

The breadboard is used for making the primary circuit connections, that is connecting the Arduino board with the L293D, the HC-05 Bluetooth Module and power sources to the Arduino and, DC motors and the IC.

For these connections connecting wires and jumper cables were used

The following image shows the circuit diagram of the actual set-up of the vehicle



Fig(6)

WORKING:

PART1: Initial steps undertaken

In this project we have used an android smartphone as a remote control which controls the vehicle via Bluetooth. This is done by" interfacing" the Arduino with the android phone

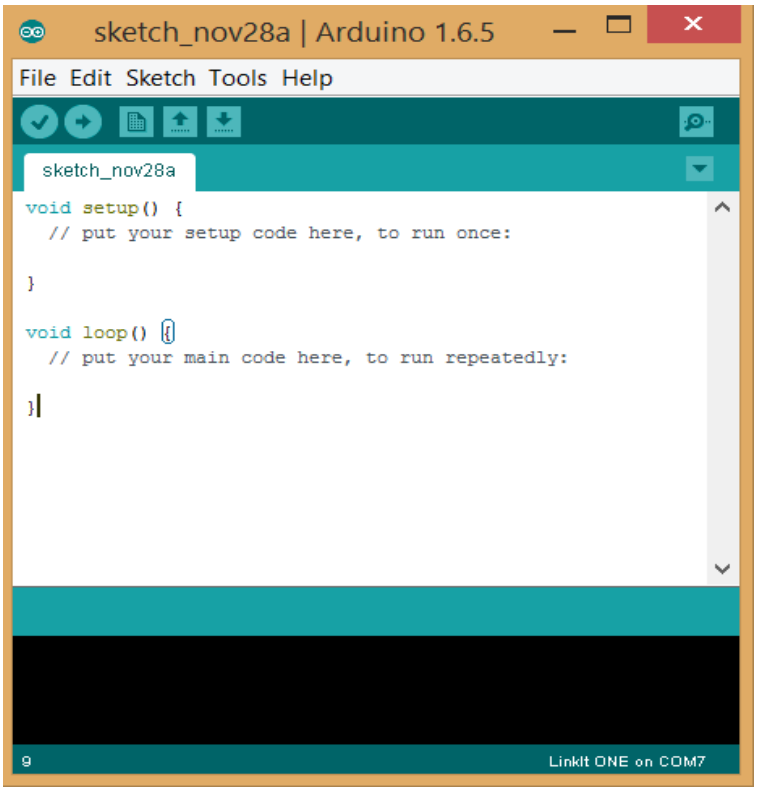
The working of this type of vehicle is fairly simple. This takes place in the following steps

1) The smartphone is connected with the vehicle via Bluetooth using the phones Bluetooth as transmitter and the HC-05 Bluetooth module connected with Arduino as a receiver.
2) The android application checks whether both the transmitter (Smartphone) and receiver (indirectly Arduino) are connectedor not.
3) Once they are connected the user can give the following instructions using the android app, they are:a) FRONT b) BACK c) LEFT d) RIGHT e) STOP.
4) The user can press any of these five buttons on the screen of the phone to see the vehicle follow the respective commands.

Once the circuit is connected in the above image, it is the next step to program the Arduino board. This is done with the help of Arduino IDE.

The basic commands in this IDE are very similar to the commands in "C++"programming. This program was downloaded for the following link.

Link:https://www.arduino.cc/en/Main/Software



The following window first appears when the Arduino application is opened .The following code was written in this command screen for the vehicle to work

CODE:

```
#include<SoftwareSerial.h>

int motor1Pin1 = 3; // pin 2 on L293D IC
int motor1Pin2 = 4; // pin 7 on L293D IC
int enable1Pin = 6; // pin 1 on L293D IC
int motor2Pin1 = 8; // pin 10 on L293D IC
int motor2Pin2 = 9; // pin 15 on L293D IC
int enable2Pin = 11; // pin 9 on L293D IC

char state;


int flag=0;       //makes sure that the serial only prints once the state
intstateStop=0;
void setup() {
   // sets the pins as outputs:
pinMode(motor1Pin1, OUTPUT);
pinMode(motor1Pin2, OUTPUT);
pinMode(enable1Pin, OUTPUT);
pinMode(motor2Pin1, OUTPUT);
pinMode(motor2Pin2, OUTPUT);
pinMode(enable2Pin, OUTPUT);
   // sets enable1Pin and enable2Pin high so that motor can turn on:
digitalWrite(enable1Pin, HIGH);
digitalWrite(enable2Pin, HIGH);
   // initialize serial communication at 9600 bits per second:
Serial.begin(9600);
}

void loop() {
   //if some date is sent, reads it and saves in state
if(Serial.available() > 0){
```

```
char state = Serial.read() ;

flag=0;

     }

   // if the state is '1' the DC motor will go forward
if (state  == '1') {
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
if(flag == 0){
Serial.println("Go Forward!");
flag=1;
       }
   }
   // if the state is '2' the motor will turn left
else if (state == '2') {
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, HIGH);
digitalWrite(motor2Pin2, LOW);
if(flag == 0){
Serial.println("Turn LEFT");
flag=1;
       }
   }

   // if the state is '3' the motor will Stop
else if (state == '3') {
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
```

```
if(flag == 0){
Serial.println("STOP!");
flag=1;
      }
stateStop=0;
   }
   // if the state is '4' the motor will turn right
else if (state == '4') {
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, HIGH);
if(flag == 0){
Serial.println("Turn RIGHT");
flag=1;
      }
 }
   // if the state is '5' the motor will Reverse
else if ( state == '5') {
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, HIGH);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
if(flag == 0){
Serial.println("Reverse!");
flag=1;
      }
  }
    //For debugging purpose
 //Serial.println(state);

}
```

Once the code is typed, hit the compile button to check if there are any warnings and errors in the code.

Once it is compiled correctly, connect the Arduino with the computer to upload the code on the board. Hit the upload button to do so.

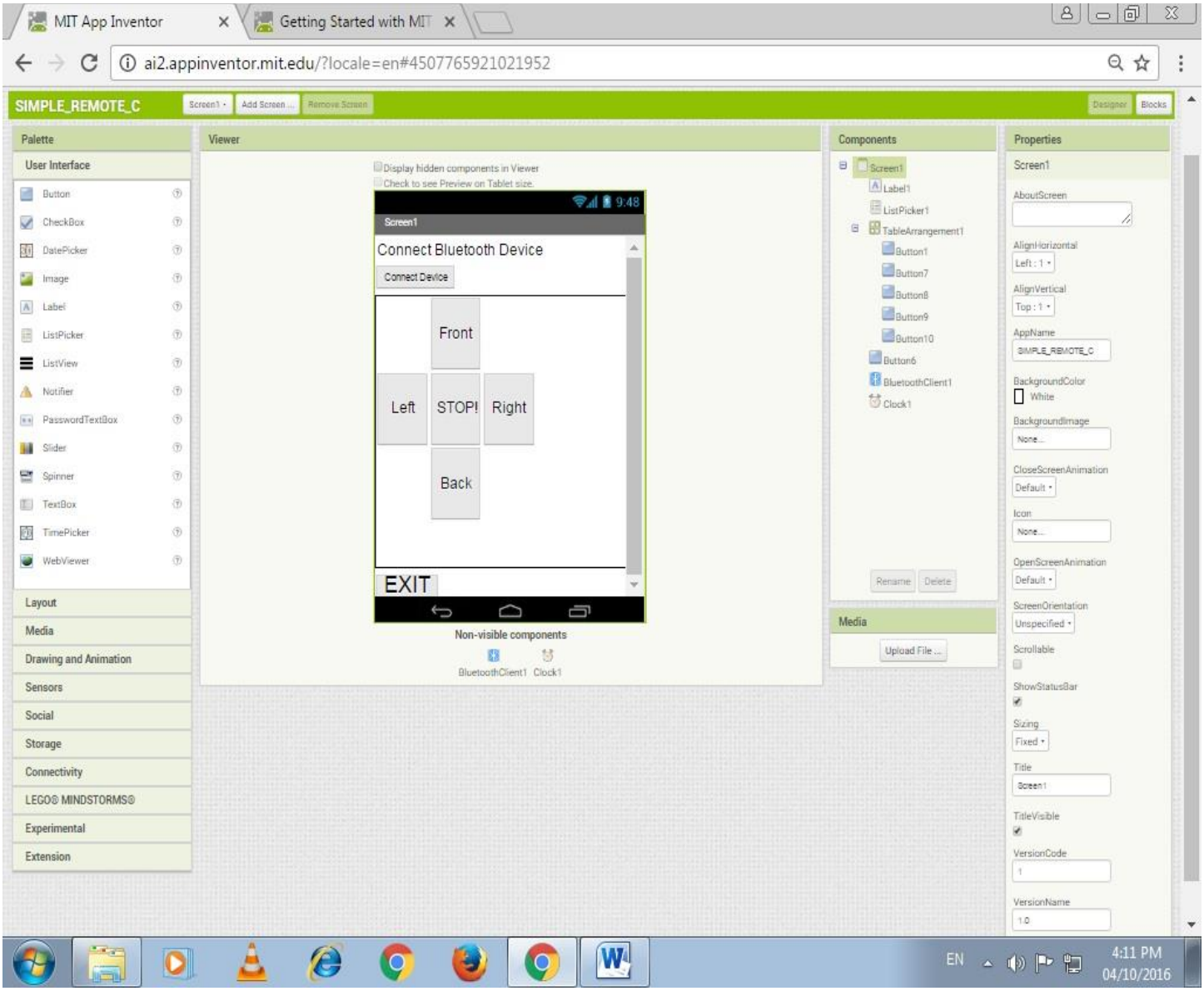
Important think to note while uploading the is, when the TX and RX pins are connected on the Arduino , the compiler would not allow the computer to upload the code. So make sure that the TX and RX pins are removed from the board while uploading the code.

PART 2:The Android application used to control the vehicle

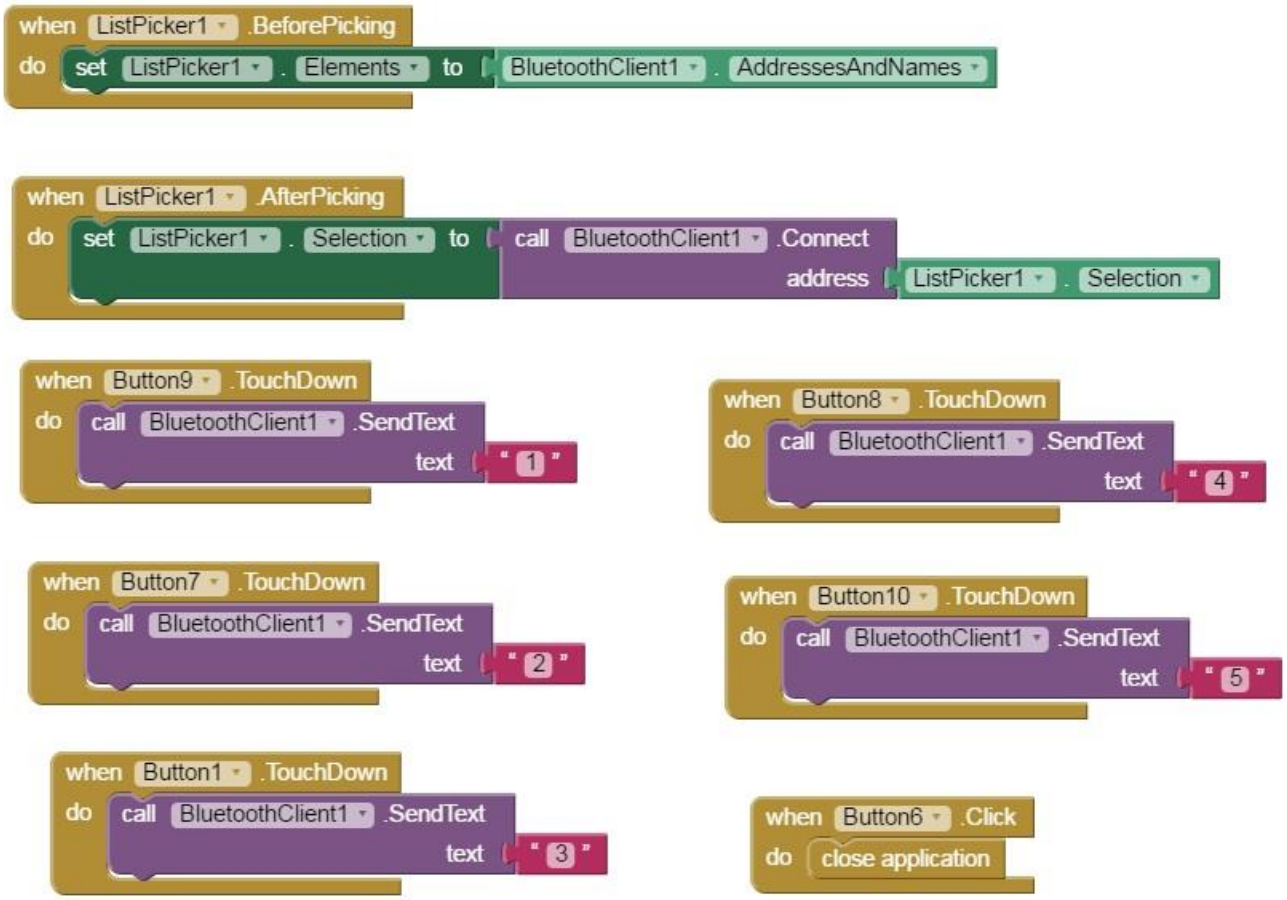
There are many android applications available on the Google Play Store for this purpose. But we did not use any of those apps for this purpose.
We developed a custom app for our project using the MitAppInventor.
This software is open source software. Anybody can use this software to make their custom android applications, like in our case to control a robot using Smartphone. To develop app inMitAppInventor one must login to their website using their Gmail accounts and now they can also make theirown applications. Using MitAppInventoris fairly easy since not much coding background is requiredand it is very easy to use. This link is provided for making an account to use MitAppInventor.

Link:http://appinventor.mit.edu/explore/get-started.html

The following image shows the basic user interface or layout of the android application.



The next image is the code that was written for making the application and sending data to the Arduino via Bluetooth



Now, once the code is uploaded and the android application is installed in the smartphone, we are ready to make the vehicle actually move using our phone.
The actual working takes place in the following way:

Let us assume that the user has clicked the “FRONT” button on his/her phone. The following mechanism is triggered:

1) In the android application the phone checks which condition was satisfied when “FRONT” button was pressed, in this case first condition was satisfied.
2) Then the android app sends a unique character value i.e. (‘1’) in this case to the Arduino via Bluetooth.

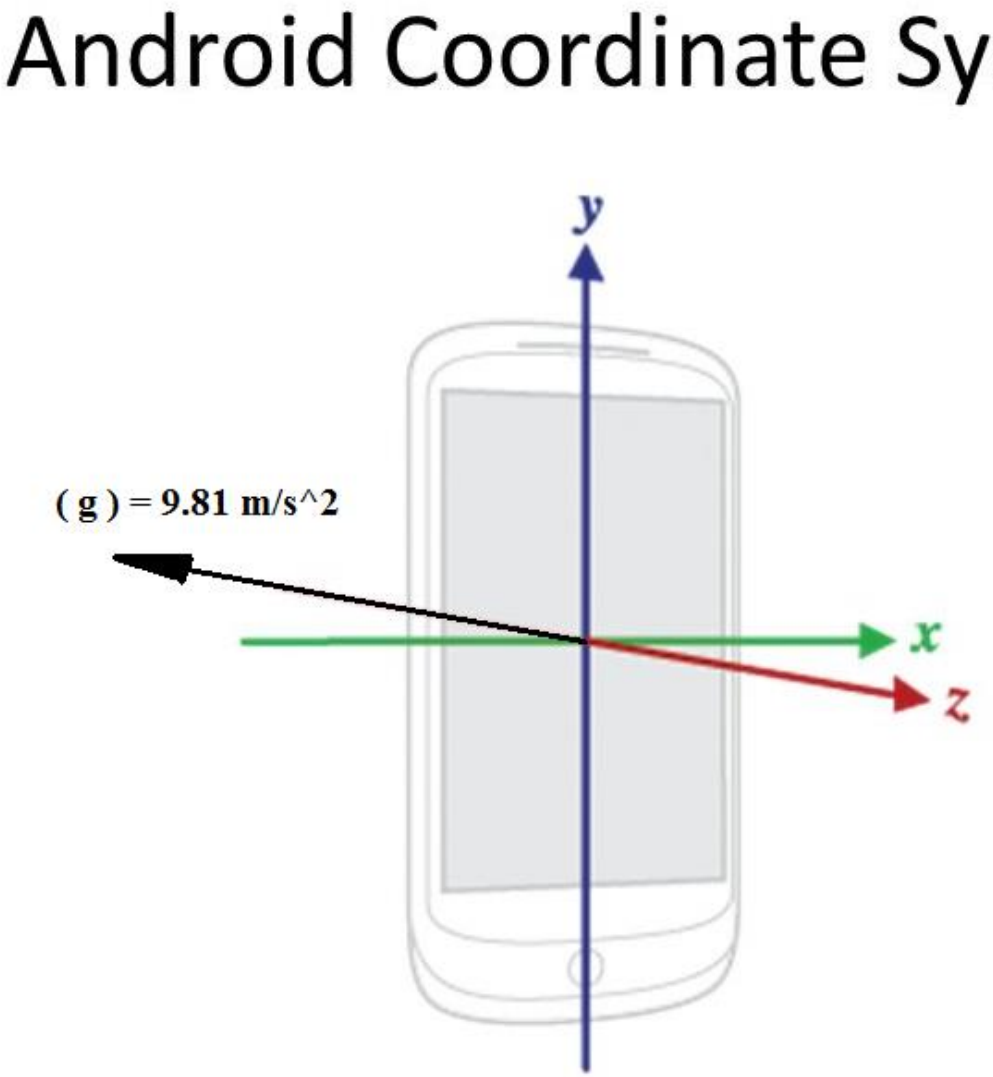
3) The Arduino code receives the value ‘1’ and gets stord in the variable defined by us in the code of Arduino called “state”.
4) The code checks the condition satisfied by the received value,and then sets the respective pin numbers on the Arduino board depending on the command received, in this case for the FRONTcommand pins will be set to high and pins will be set low .
5) This high and low is the voltage values, which will be supplied to the DC-Motors by the IC-L293D and the vehicle will follow the command as its motors rotate, i.e. Motor1 clockwise and Motor2 anticlockwisein this case

Similar mechanism takes place when the rest of the buttons are pressed i.e. BACK, RIGHT, LEFT, STOP, and depending on the command the vehicle will move accordingly.

The next part of our project is under initial stages of working and we are able to control the robot using gestures of our smartphone. But there are bugs and design fixes needed for this part of the project to work. The robot followsa particular set of instructions but not after a certain amount.

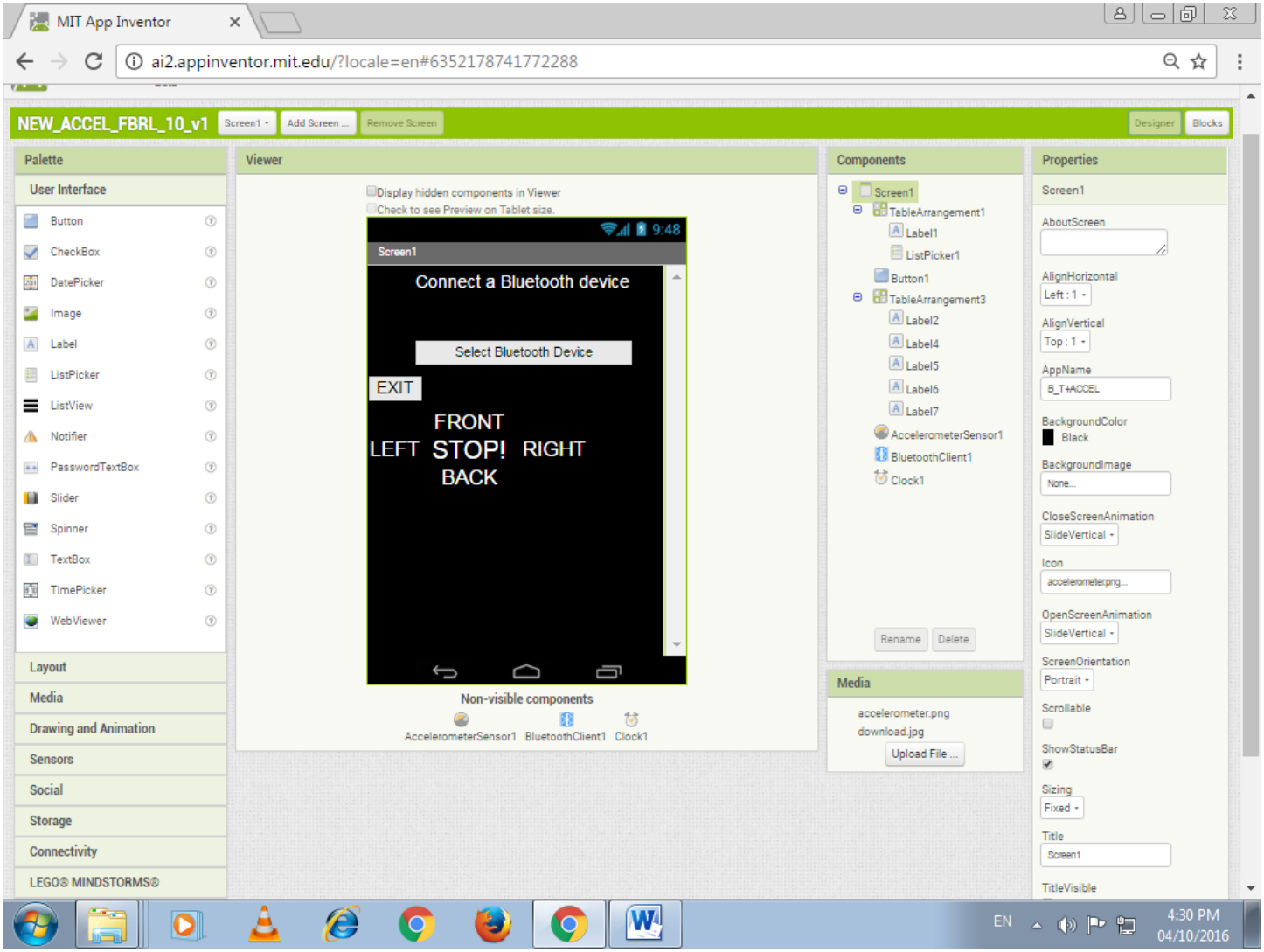
PART C: Controlling the robot by using Gestures of smartphone

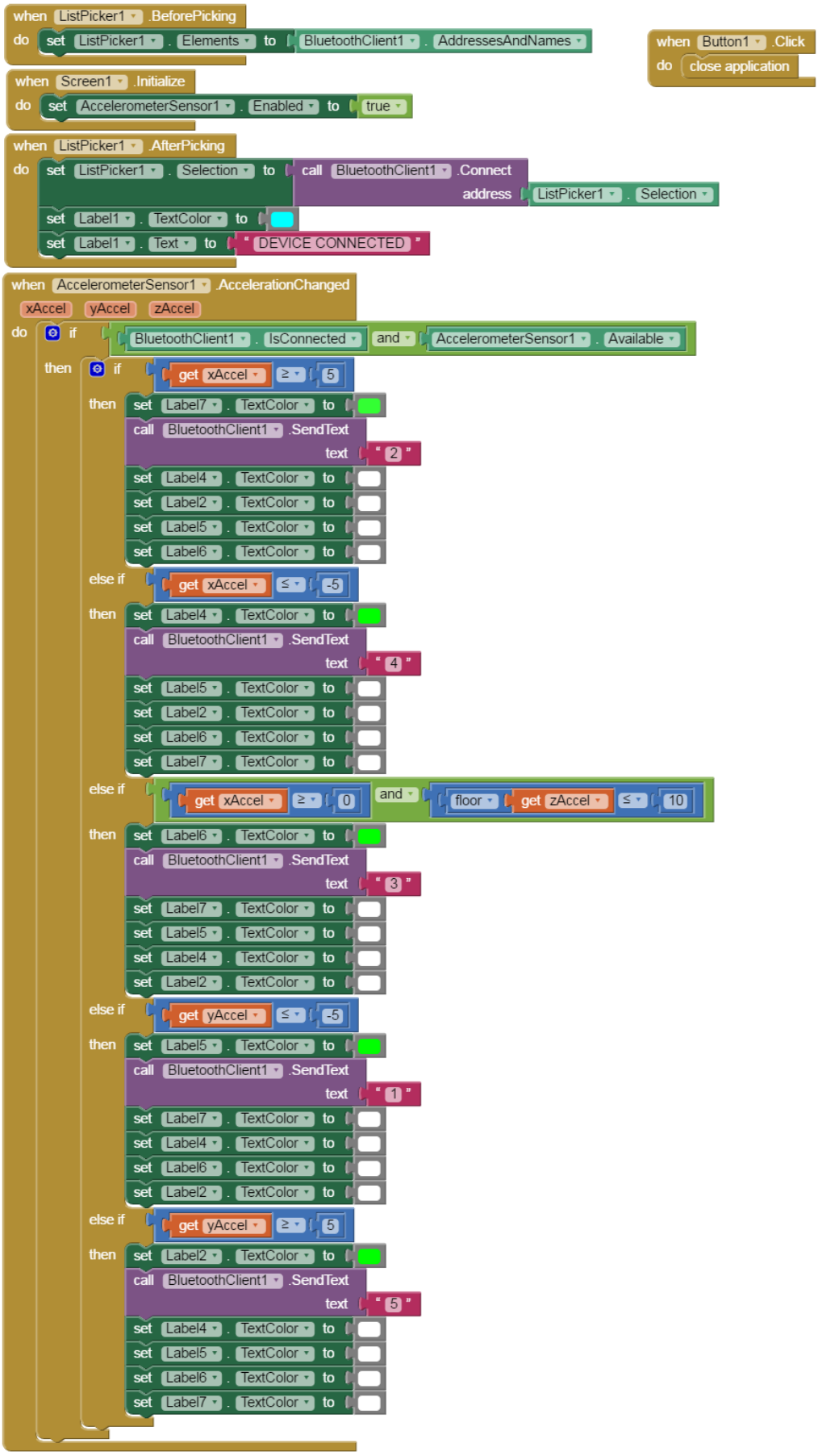
The coding in the Arduino is the same as that of the above project.
The only change is to be made in the coding of the android application.
Now-a-days, almost all the smartphones have all kinds of sensors. For this project we use the Accelerometer that is inbuilt in the phone to control the robot.
The robot is controlled by tilt gestures of the phone. The following image shows the orientation of the X, Y and Z axes of the phone, when it is placed flat facing up on table parallel to the surface of the earth. In this position the value of acceleration is on the Z axis



Now when the phone is tilted to the LEFT, the value of acceleration due to gravity (g) starts to concentrate on the X axis, and the more the tilt the closer it is to the value of “g”. Similarly for RIGHT negative X axis approaches “g”.
Similarly when the phone is tilted towards FRONT the value of “g”, starts to concentrate on the Y axis, also the more the tilt the closer it is to the value of “g”.”. Similarly for BACK negative Y axis approaches “g”.
When the value of “g” is concentrated on the Z axis only, consider it to be in the stop position.

The following shows the user interface and coding of the android application:




when ListPicker1 .BeforePicking
do set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames
when Button1 .Click
do close application
when Screen1 .Initialize
do set AccelerometerSensor1 . Enabled to true
when ListPicker1 .AfterPicking
do set ListPicker1 . Selection to call BluetoothClient1 .Connect
address ListPicker1 . Selection
set Label1 . TextColor to
set Label1 . Text to " DEVICE CONNECTED "
when AccelerometerSensor1 .AccelerationChanged
xAccel yAccel zAccel
do if BluetoothClient1 . IsConnected and AccelerometerSensor1 . Available
then if get xAccel ≥ 5
then set Label7 . TextColor to
call BluetoothClient1 .SendText
text " 2 "
set Label4 . TextColor to
set Label2 . TextColor to
set Label5 . TextColor to
set Label6 . TextColor to
else if get xAccel ≤ -5
then set Label4 . TextColor to
call BluetoothClient1 .SendText
text " 4 "
set Label5 . TextColor to
set Label2 . TextColor to
set Label6 . TextColor to
set Label7 . TextColor to
else if get xAccel ≥ 0 and floor get zAccel ≤ 10
then set Label6 . TextColor to
call BluetoothClient1 .SendText
text " 3 "
set Label7 . TextColor to
set Label5 . TextColor to
set Label4 . TextColor to
set Label2 . TextColor to
else if get yAccel ≤ -5
then set Label5 . TextColor to
call BluetoothClient1 .SendText
text " 1 "
set Label7 . TextColor to
set Label4 . TextColor to
set Label6 . TextColor to
set Label2 . TextColor to
else if get yAccel ≥ 5
then set Label2 . TextColor to
call BluetoothClient1 .SendText
text " 5 "
set Label4 . TextColor to
set Label5 . TextColor to
set Label6 . TextColor to
set Label7 . TextColor to

Reference:

1) http://www.instructables.com/id/Simple-RC-car-for-beginners-Android-control-over-/?ALLSTEPS
2) http://www.instructables.com/id/Accelerometer-bluetooth-controlled-Mini-Car/?ALLSTEPS
3) http://appinventor.mit.edu/explore/get-started.html
4) https://forum.arduino.cc/
5) https://developer.android.com/guide/topics/sensors/sensors_motion.html