PROJECT REPORT

DSE

National Institute of Business Management

Group Members:
AHAMED - CODSE172F-012
PASINDU - CODSE172F-023
PADMAKANTH - CODSE172F-032
DEVIN - CODSE172F-051

Table of Contents

Introduction

Today the world is moving towards automated robots and AI technologies which replace human beings for multi-task activities. The essence of learning and applying state-of-art technologies is paramount to upgrade the living standards. To facilitate the above ideology, the project “Automated garbage collecting robot (Garby)” was initiated.

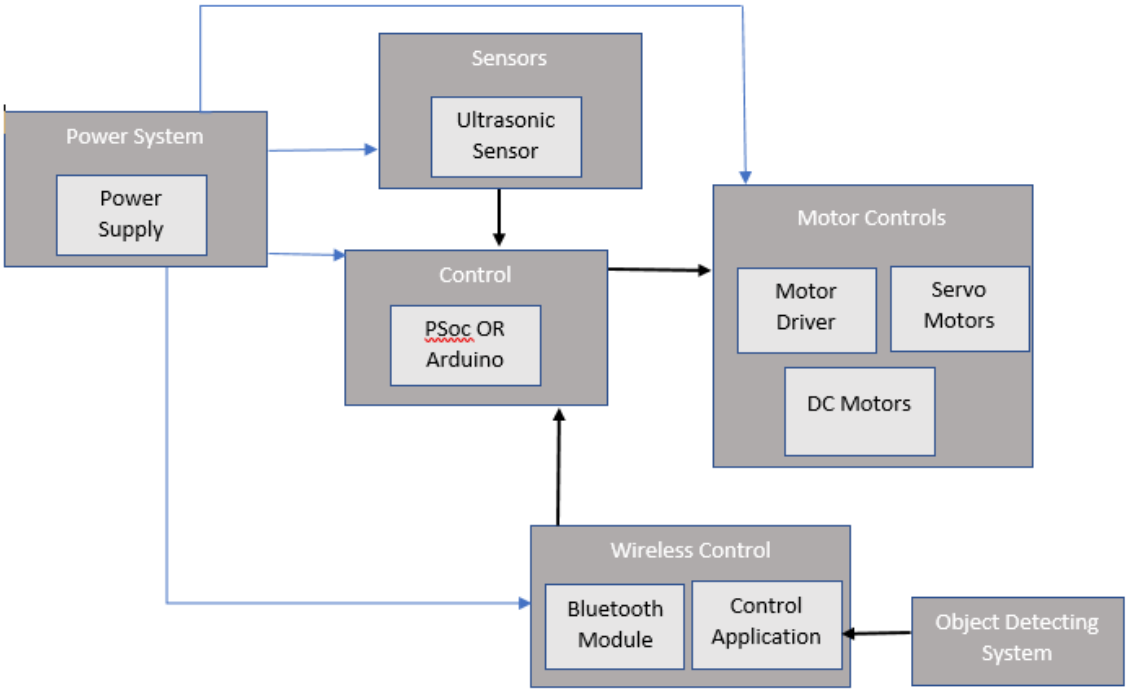
The project comprises Kinematics, Control, Embedded Applications and Image Processing. The machine learning or AI parts, path finding algorithms and ROS (Robotic Operating Systems) are waiting to be complete. In the first step, the algorithms necessary for activating basic functions would be accomplished.

Conclusively, the project scope is focused on building a working prototype with the basic functions which includes basic movements of robot and the arm, image processing task and fundamental embedded system development.

Hardware Design

Block Diagram

Our design consists of six main modules power system, sensors, control, motor control, wireless control and object detecting system. The power system provides 12V to the DC motors,7V to servo motors and 5V to the rest of the system. Sensors include an ultrasonic sensor. The PSoC is in the center taking both sensor and wireless inputs and outputting to the motors. Motor control consists motor controller DC motors & servo motors. Finally, wireless control includes the Bluetooth module as well as the phone application and its communication protocol.

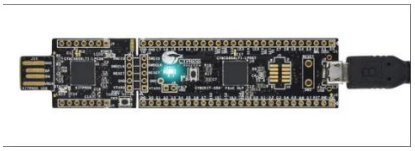


Block diagram

Control

PSoC

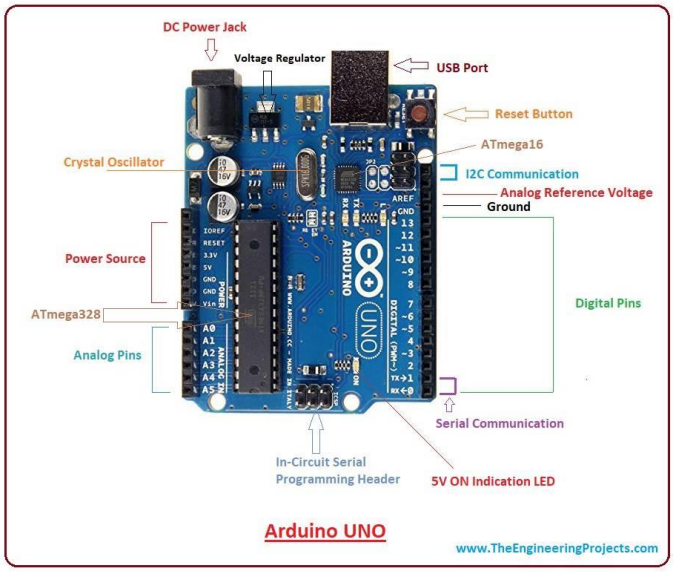
PSoC (programmable system-on-chip) is a family of microcontroller integrated circuits by Cypress Semiconductor. These chips include a CPU core and mixed-signal arrays of configurable integrated analog and digital peripherals.



Arduino

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.

The Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output pins that may be interfaced to various expansion boards and other circuits.



Power System

Power Supply

This module consists of AC to DC converter(12V), power bank(5V) and voltage regulator (12V to 7V).5V DC output supply to the PSoC, Bluetooth module and ultrasonic sensor.12V DC voltage supply to the motor driver and DC motors.7V DC voltage supply to the servo driver.

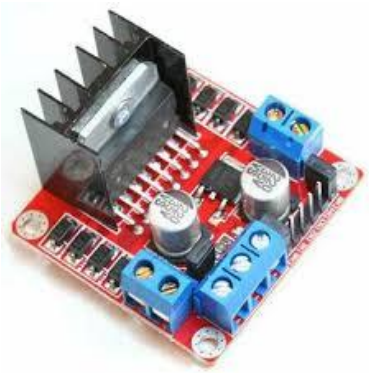
Motor Control

Servo Motors

A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback.

Motor Driver l298N Dual

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors.



DC Motors

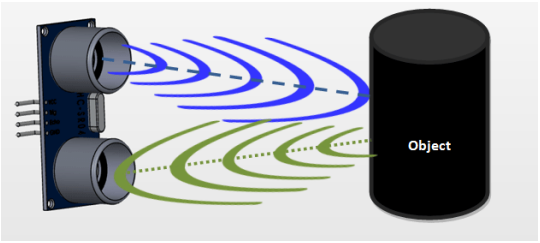
A DC motor is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields.

Sensors

Ultrasonic Sensor

Ultrasonic sound vibrates at a frequency above the range of human hearing. ... Our **ultrasonic sensors**, like many others, use a single **transducer** to send a pulse and to receive the echo. The **sensor** determines the distance to a target by measuring time lapses between the sending and receiving of the **ultrasonic** pulse.

Ultrasonic sensors work by emitting **sound waves** at a frequency too high for humans to hear. They then wait for the **sound** to be reflected back, calculating **distance** based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object.

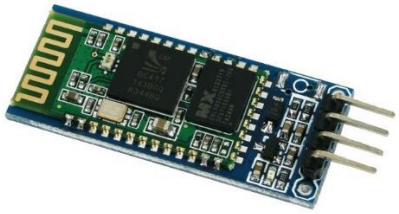


Wireless Controls

HC-06 Bluetooth Module

The **HC-06** is a class 2 slave **Bluetooth module** designed for transparent wireless serial communication. Once it is paired to a master **Bluetooth** device such as PC, smart phones and tablet, its operation becomes transparent to the user.

The BT Bluetooth module is a stackable shield with serial ports based on the HC-06 module. The shield can be connected directly to the Arduino UART port for wireless communication. Without obstacles or other interference, the Bluetooth shield can communicate in a range of **10 meters** (**32ft**).



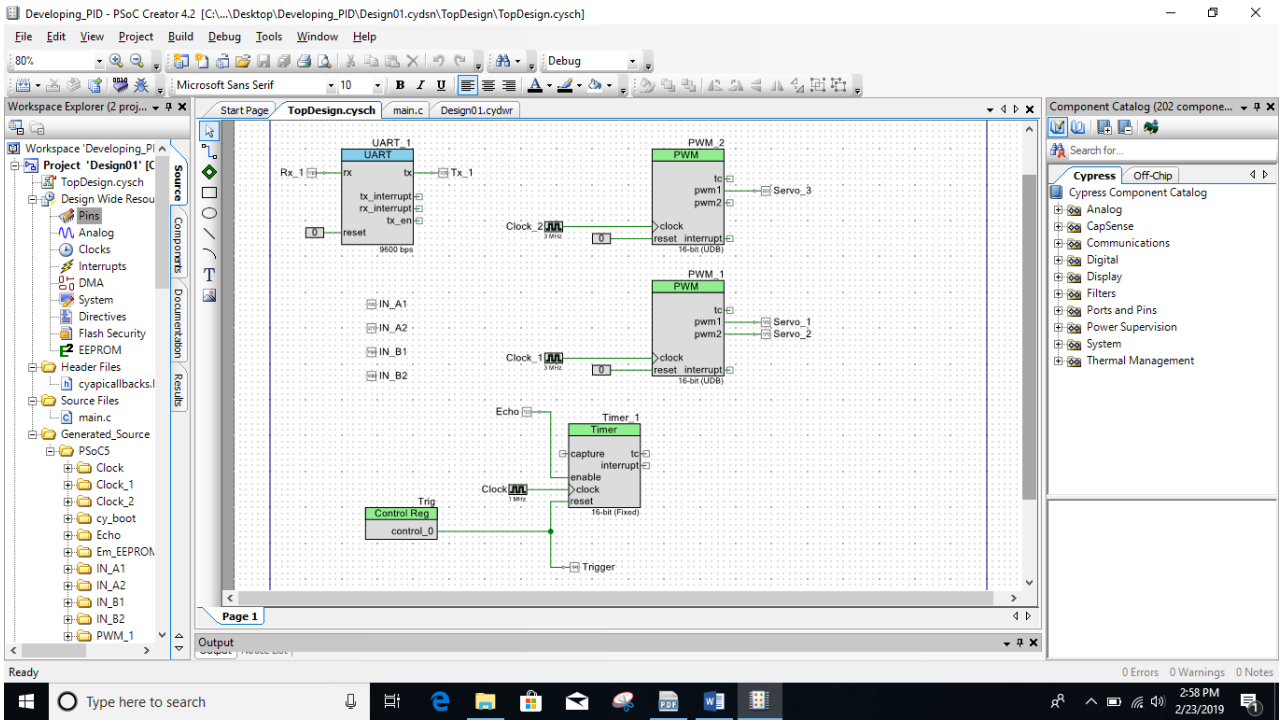
Object Detecting System

Mobile Phone

A **camera phone** is a **mobile phone** which is able to capture photographs and often record video using one or more built-in digital **cameras**. ... Most **camera** phones are simpler than separate digital **cameras**. Their usual fixed-focus lenses and smaller sensors limit their performance in poor lighting.

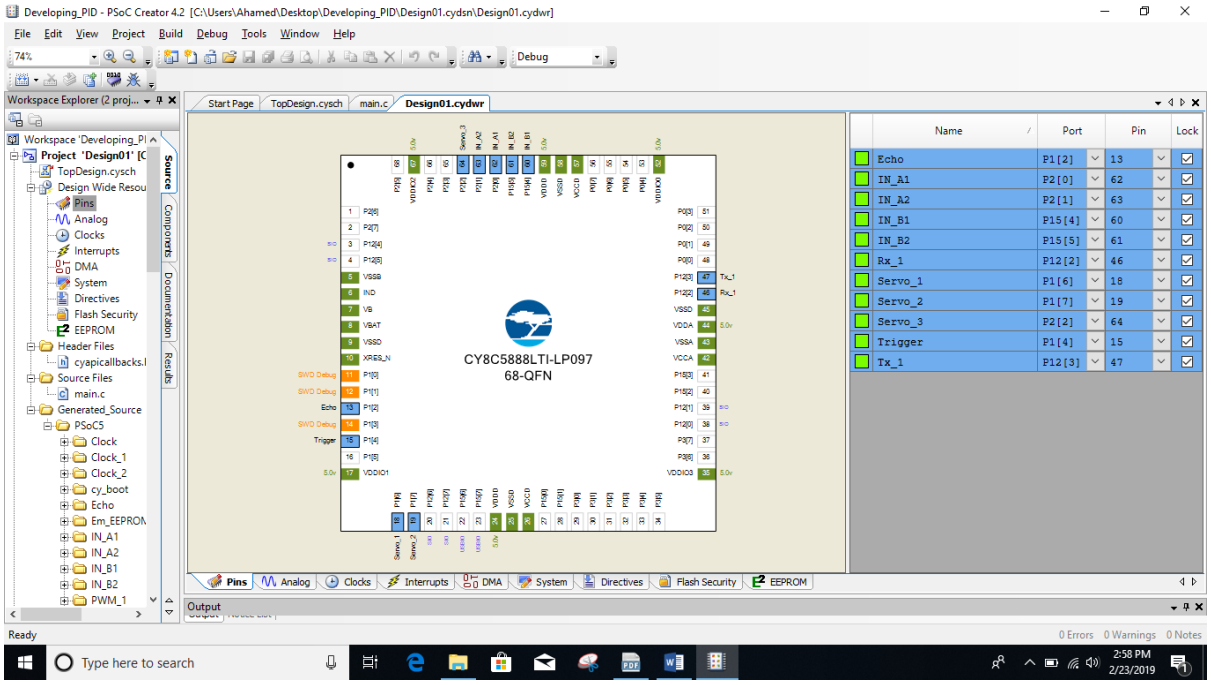
Mobil camera use to detect the image

Hardware Design in PSoC Creator



Connecting Modules Togather

PSoC



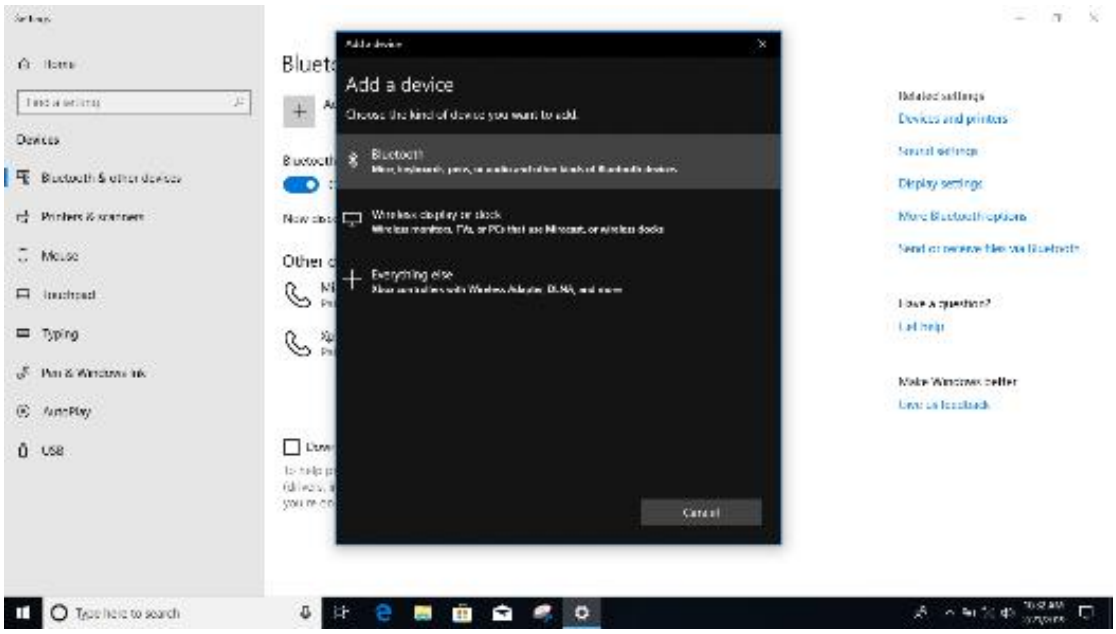
Module	Pins	Connect to
HC-06	GND	Ground
	VCC	5V
	TX	12.2 pin in Psoc
	RX	12.3 pin in PSoc
Ultrasonic Sensor	VCC	5V
	Trig	1.4
	Echo	1.2
	GND	Ground
Motor Driver l298N Dual	VCC	12V
	GND	Ground
	5V	
	OUT1	DC motor input
	OUT2	DC motor input
	OUT3	DC motor input
	OUT4	DC motor input
Servo Motors	Signal (Yellow)	PWM
	VCC (Red)	7V
	GND (Black)	Ground

Software Design

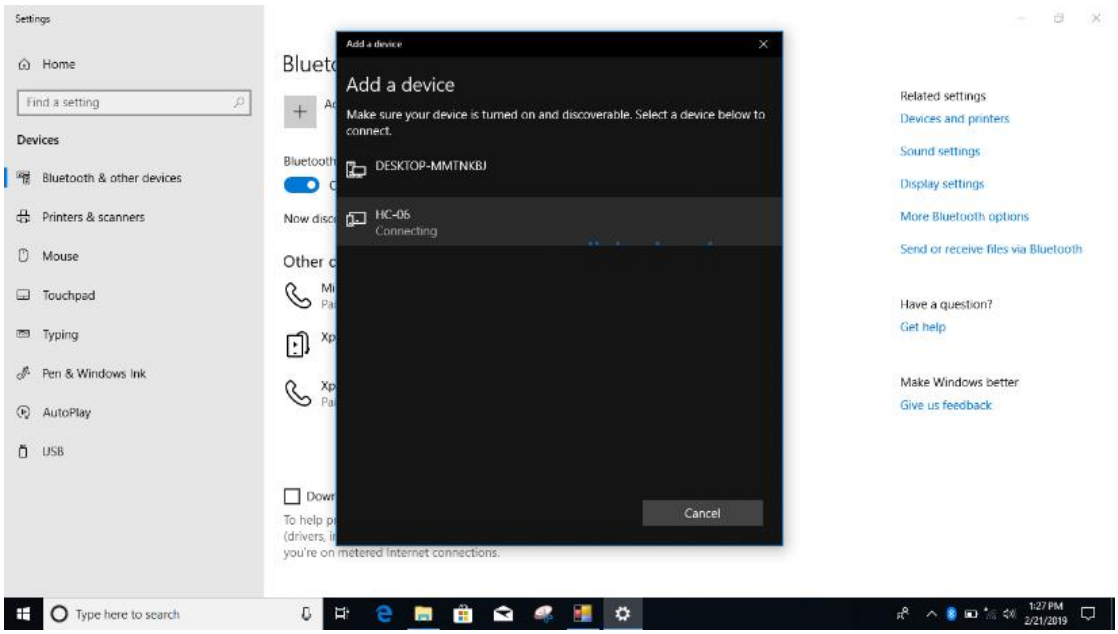
Pairing Bluetooth module HC-06

Once the module is configured as you wish, you can pair the module HC-06 to the device of your choice like any Bluetooth device. Select the name of your module in the list of available Bluetooth device (default is HC-06) and enter the PIN code (default is 1234). When it is done, The LED should stop blinking.

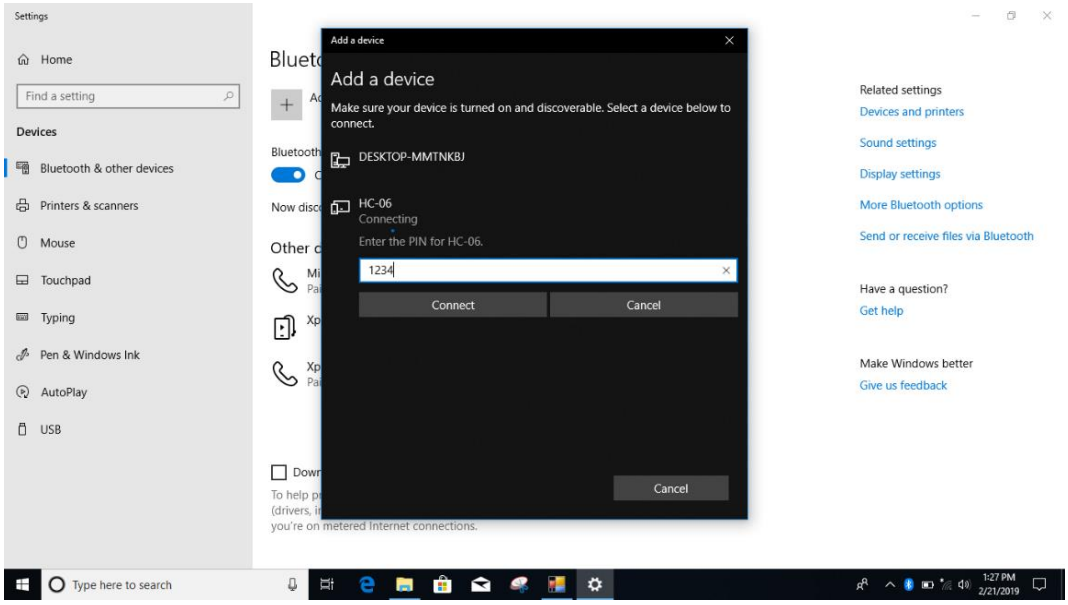
1. Search the avaliable boletooth devices



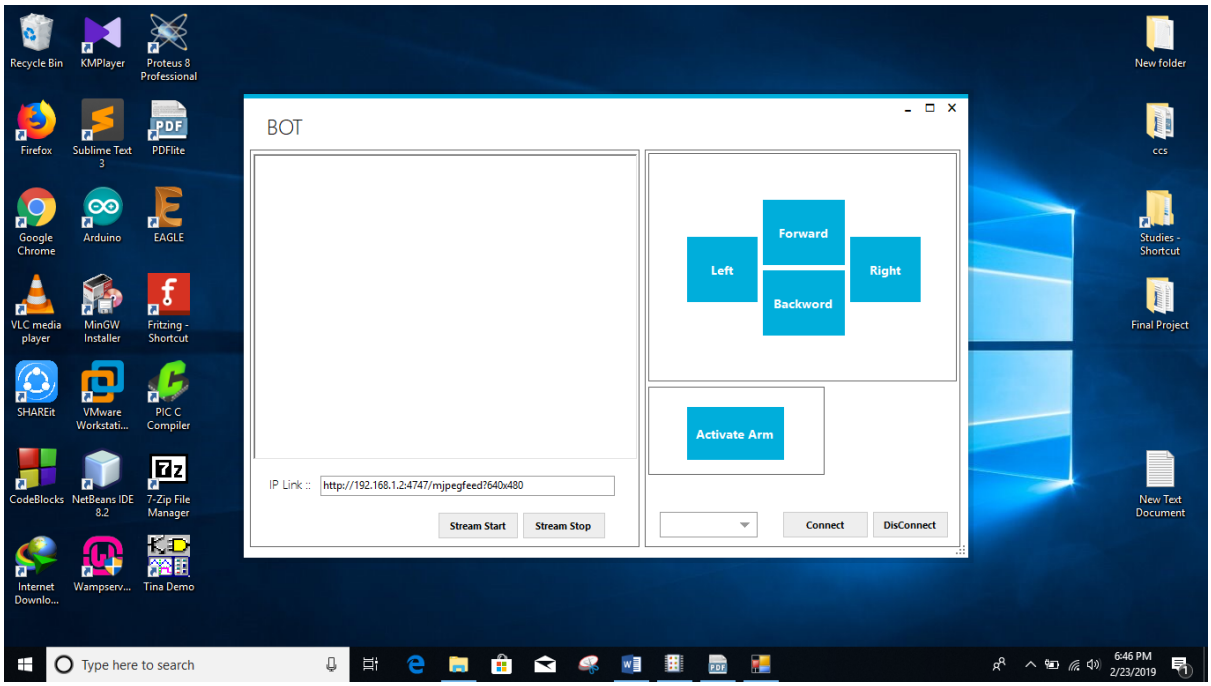
2. Select the hc-06 bluetooth module



3. Enter the default password of hc-06(1234 or 0000) and connect it.



Control Application



Button	Functionality	Passing Character as bluetooth command
Connect	Connect to Bluetooth Device through COM port	
DisConnect	Disconnect the Bluetooth connection	
Forward	Move forward	C
Backward	Move backward	B
Left	turn left	D
Right	Turn right	E
Activate Arm	Activate arm to follow the generated instruction	A
Stream start	Start the video streaming	
Stream stop	Stop the streaming	

Source code

PSoC

```
#include "project.h"
#include "math.h"
#include "stdio.h"

float rev;
char strMsg[50];

float X0 = 10;               //Assign values for coordinates
float Y0 = 17;
float Z0 = 9;

float a1 = 10;              //Lengths of links
float a2 = 10;
float a3 = 7.5;

float ak = 8;              //Length and angle of preset
float Tk = 0;

int T0,T1,T2;

float deg2rad(float deg){

    float rad = (deg/180)*3.14;

    return rad;
}

float rad2deg(float rad){

    float deg = (rad/3.14)*180;

    return deg;
}

void calculate(){

    float r1 = (pow((pow(X0,2)+pow(Y0,2)),(1./2))-ak*cos(Tk));  //Calculate
triangle distances
    float r2 = a1+ak*sin(Tk)-Z0;
    float r3 = pow((pow(r1,2)+pow(r2,2)),(1./2));

    float e1 =(pow(a3,2)-pow(a2,2)-pow(r3,2))/(-2*a2*r3);
    float e2 =(pow(r3,2)-pow(a2,2)-pow(a3,2))/(-2*a2*a3);

    float G1 = acos(e1);             //Calculate triangle angles
    float G2 = atan(r2/r1);
    float G3 = acos(e2);

    T0 = rad2deg(atan(Y0/X0));    //Calculate required angles to move
    T1 = rad2deg(G2)-rad2deg(G1);
    T2 = 180 - rad2deg(G3);

    sprintf(strMsg,"%d",T0);
```

```
    UART_1_PutString(strMsg);
    UART_1_PutString("++");

    sprintf(strMsg,"%d",T1);
    UART_1_PutString(strMsg);
    UART_1_PutString("++");

    sprintf(strMsg,"%d",T2);
    UART_1_PutString(strMsg);
    UART_1_PutString("++");

}


int calAngle(float ang){
    // EQUTATION: y = ((ymax - ymin)/(xmax - xmin))*(x-xmin)+ymin;

    int com = ((7000-2800)/(180-0))*(ang-0)+2800;
    return com;

}

void armcontrol(){

    PWM_2_WriteCompare1(calAngle(T0));
    CyDelay(1000);
    PWM_1_WriteCompare1(calAngle(T1));
    CyDelay(1000);
    PWM_1_WriteCompare2(calAngle(T2));
    CyDelay(3000);


    PWM_1_WriteCompare1(calAngle(0));
    CyDelay(1000);
    PWM_1_WriteCompare2(calAngle(0));
    CyDelay(1000);
    PWM_2_WriteCompare1(calAngle(100));
    CyDelay(1000);

}

void distance(){

    while(Echo_Read() == 0){

        Trig_Write(1);
        CyDelay(10u);
        Trig_Write(0);
        CyDelay(1);

     }
     while(Echo_Read() == 1){}

        int count_pulse = 65535 - Timer_1_ReadCounter();
        int distance = count_pulse/58.0;
        Z0 = 18.9 - distance;
        sprintf(strMsg,"%d",distance);
        UART_1_PutString(strMsg);
        UART_1_PutString("++");

}
```

```
void stop(){

    IN_A1_Write(0);
    IN_A2_Write(0);
    IN_B1_Write(0);
    IN_B2_Write(0);

}

int main(void)
{

    Timer_1_Start();
    UART_1_Start();
    PWM_1_Start();
    PWM_2_Start();

    uint8 recieve;

    Tk = deg2rad(20);

    for(;;)
    {
        recieve = UART_1_GetChar();

        if(recieve == 'A'){

            stop();
            distance();
            calculate();
            armcontrol();

            recieve = 0;

        }else if(recieve == 'B'){    //Forward

            IN_A1_Write(1);
            IN_A2_Write(0);
            IN_B1_Write(1);
            IN_B2_Write(0);

            CyDelay(250);
            //stop();

            recieve = 0;


        }else if(recieve == 'C'){    // Backward

            IN_A1_Write(0);
            IN_A2_Write(1);
            IN_B1_Write(0);
            IN_B2_Write(1);

            CyDelay(250);
            //stop();

            recieve = 0;
```

```
        }else if(recieve == 'D'){    //Left

            IN_A1_Write(0);
            IN_A2_Write(1);
            IN_B1_Write(1);
            IN_B2_Write(0);

            CyDelay(250);
            //stop();

            recieve = 0;

        }else if(recieve == 'E'){    //Right

            IN_A1_Write(1);
            IN_A2_Write(0);
            IN_B1_Write(0);
            IN_B2_Write(1);

            CyDelay(250);
            //stop();

            recieve = 0;

        }else if(recieve == 'Z'){

            stop();
            recieve = 0;

        }

    }
}
```

Arduino

```
#include <SoftwareSerial.h> // Attaching required libraries
#include <Servo.h>
#include <math.h>

SoftwareSerial BTSerial(2, 3);

Servo myservo1;          //servo declaration
Servo myservo2;
Servo myservo3;

float X0 = 2;           //Assign values for coordinates
float Y0 = 4;
float Z0 = 4;

float a1 = 6.5;          //Lengths of beams
float a2 = 6.5;
float a3 = 6.5;

float ak = 4;           //Length and angle of preset beams(ak - angular beam)
float Tk = 90;

int T0, T1, T2;          // angles for base,elbow and wrist respectively

_Bool isCal = 0;

const int trig = 7;       // Assigning pins for ultra sonic sensor
const int echo = 8;
```

```
char A = '0';

float deg2rad(float deg) {  // degrees to radians

  float rad = (deg / 180) * 3.14;

  return rad;
}

float rad2deg(float rad) {  // radians to degrees

  float deg = (rad / 3.14) * 180;

  return deg;
}

void distance() {   // calculate distance

  digitalWrite(trig, HIGH);
  delayMicroseconds(20);
  digitalWrite(trig, LOW);
  float duration = pulseIn(echo, HIGH);
  float distance = duration * 0.034 / 2;
  Z0 = 13.4 - distance; // calculate height of the object
  Serial.print("height of the object ");
  Serial.println(Z0);

}

void calculate() {  // calculate angles theta 1,2,3
```

```
  float r1 = (pow((pow(X0, 2) + pow(Y0, 2)), (1. / 2)) - ak * cos(Tk)); //Calculate triangle distances
  float r2 = a1 + ak * sin(Tk) - Z0;
  float r3 = pow((pow(r1, 2) + pow(r2, 2)), (1. / 2));

  Serial.println("r values");
  Serial.println(r1);
  Serial.println(r2);
  Serial.println(r3);

  float e1 = (pow(a3, 2) - pow(a2, 2) - pow(r3, 2)) / (-2 * a2 * r3);
  float e2 = (pow(r3, 2) - pow(a2, 2) - pow(a3, 2)) / (-2 * a2 * a3);

  Serial.println("e values");
  Serial.println(e1);
  Serial.println(e2);


  float G1 = acos(e1);        //Calculate triangle angles
  float G2 = atan(r2 / r1);
  float G3 = acos(e2);

  Serial.println("G values");
  Serial.println(G1);
  Serial.println(G2);
  Serial.println(G3);

  T0 = rad2deg(atan(Y0 / X0));  //Calculate required angles to move
  T1 = rad2deg(G2) - rad2deg(G1);
  T2 = 180 - rad2deg(G3);

}
```

```
void armcontrol() {   // arm control servo commands

  Serial.println("angle values");

  myservo1.write(T0); // angles for picking
  Serial.println(T0); // base  angle
  delay(500);
  myservo2.write(T1+90); // elbow angle
  Serial.println(T1);
  delay(500);
  myservo3.write(T2); // wrist angle
  Serial.println(T2);

  delay(5000);

 /* myservo1.write(0); // angles for stationary position
  delay(500);
  myservo2.write(90);
  delay(500);
  myservo3.write(0);
  delay(500);*/

}

void Stop() {

  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  digitalWrite(12, LOW);
  digitalWrite(13, LOW);
```

```
}
void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600); // Serial monitor
  pinMode(7, OUTPUT); // setting pins for ultra sonic
  pinMode(8, INPUT);

  pinMode(4, OUTPUT);  // setting pins for movements of the robot wheels
  pinMode(5, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);

  pinMode(9, OUTPUT); // setting pins for enabling bluetooth
  digitalWrite(9, HIGH);
  Serial.begin(9600);
  Serial.println("Enter AT commands:");

  BTSerial.begin(9600); // Bluetooth starting

  myservo1.attach(6);   // attaching servos for pins
  myservo2.attach(10);
  myservo3.attach(11);

}

void loop() {

  // put your main code here, to run repeatedly:
```

```
if (Serial.available() > 0) { // Reading data from serial monitor

  A = Serial.read();

}

if (BTSerial.available() > 0) { // Reading data from Bluetooth

  A = BTSerial.read();

}

Tk = deg2rad(90);

if (A == 'A') {       //Arm Control commands
  distance();
  calculate();
  armcontrol();
  A = '0';

} else if (A == 'B') {   // wheels control commands

  digitalWrite(4, HIGH); // backward
  digitalWrite(5, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(13, LOW);
  delay(150);
  Stop();

} else if (A == 'C') {   // forward
```

```
  digitalWrite(4, LOW);
  digitalWrite(5, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(13, HIGH);
  delay(150);
  Stop();

} else if (A == 'D') {   //left turn

  digitalWrite(4, HIGH);
  digitalWrite(5, LOW);
  digitalWrite(12, LOW);
  digitalWrite(13, HIGH);
  delay(150);
  Stop();

} else if (A == 'E') {   // right turn

  digitalWrite(4, LOW);
  digitalWrite(5, HIGH);
  digitalWrite(12, HIGH);
  digitalWrite(13, LOW);
  delay(150);
  Stop();

} else if (A == 'Z') {   // stop

  Stop();

}
```

}

Kinematics: mathematical model equations and how we derived it.

$$\Theta_0 = \frac{y_3^0}{x_3^0}\ (base\ motor)$$

$$\Theta_1 = tan^{-1}\left(\frac{z_3^0 - a_1 - a_k sin\Theta_k}{(x_3^0)^2 + (y_3^0)^2 - a_k cos\Theta_k}\right)\ (elbow\ motor)$$

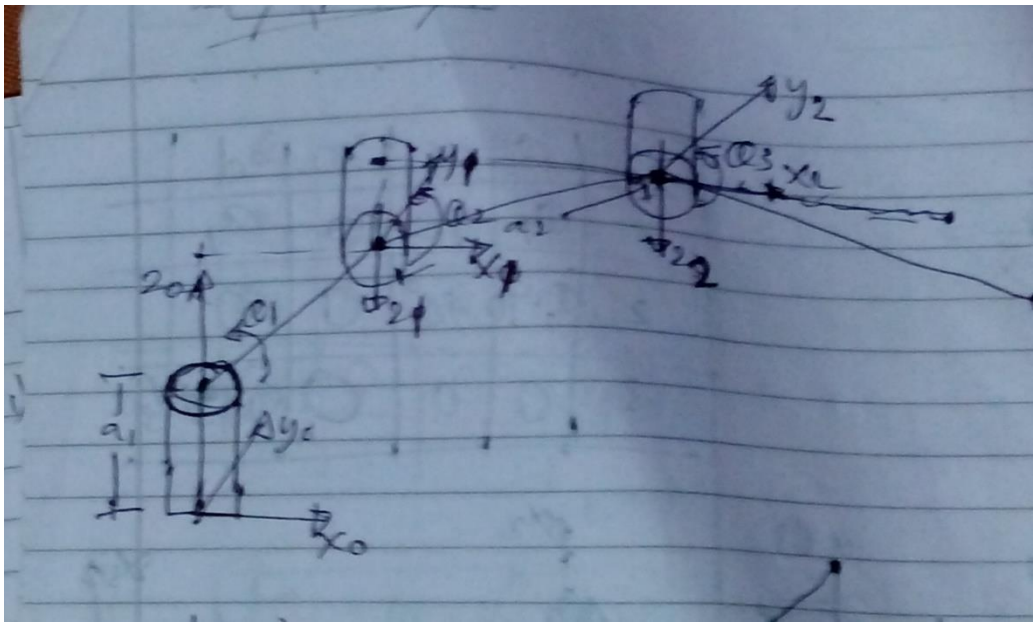
$$\Theta_2 = 180 - cos^{-1}\left(\frac{\left(\sqrt{(x_3^0)^2 + (y_3^0)^2} - a_k cos\Theta_k\right)^2 - {a_2}^2 - {a_3}^2}{-a_2 a_3}\right)(wrist\ motor)$$

The equations were typed in numpy library in python and worked successfully when $a_1 > z_3^0$.

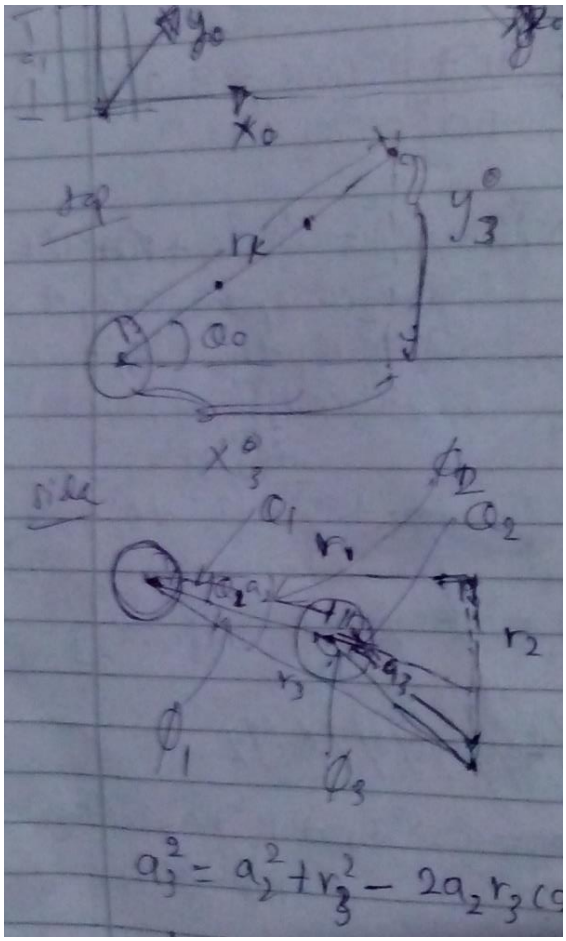
The answer seems to be negative with smaller mistake inside tan^{-1}. But it didn't affect the final answer.

Figures of hand written derivations

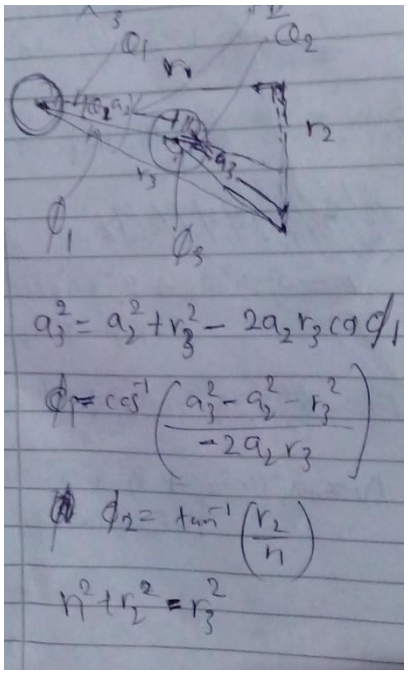
Denavit-Hartenburg frames

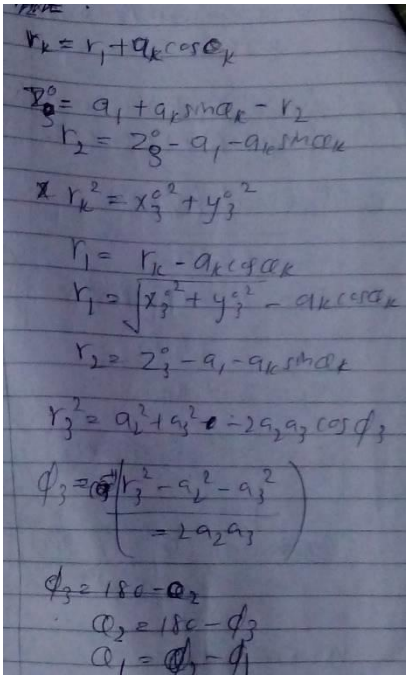


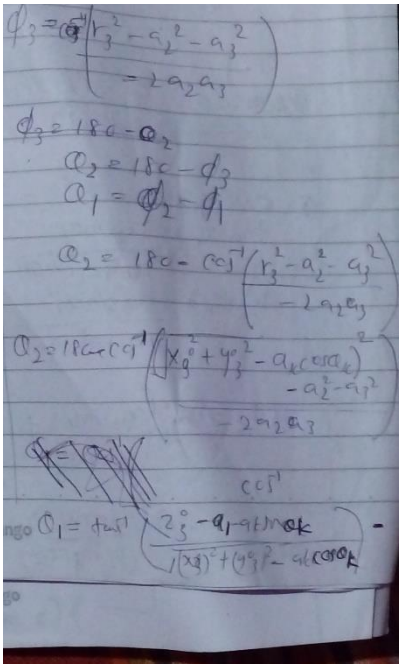
Derivation: top and side views



Derivation: making use of Pythagoras theorem and cosine rule for finding necessary angles.







For Control Application

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using AForge.Video;
using AForge.Video.DirectShow;

namespace controlapp
{
    public partial class Form1 : MetroFramework.Forms.MetroForm
    {
        MJPEGStream stream;
        public Form1()
        {

            InitializeComponent();
                //show list of valid com ports
                foreach (string port in System.IO.Ports.SerialPort.GetPortNames())
                {
                    cmbCOMP.Items.Add(port);
                }
        }

        private void stream_NewFrame(object sender, NewFrameEventArgs eventArgs)
        {
            Bitmap bmp = (Bitmap)eventArgs.Frame.Clone();
            videoBOX.Image = bmp;
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void txtIPLINK_Click(object sender, EventArgs e)
        {

        }

        private void btnSSTART_Click(object sender, EventArgs e)
        {
            try {
                //Start the video stream
                string iplink = txtIPLINK.Text;
                stream = new MJPEGStream(iplink);
                stream.NewFrame += stream_NewFrame;
                stream.Start();
            }catch(Exception ex)
            { MessageBox.Show(ex.Message);}
        }

        private void btnSSTOP_Click(object sender, EventArgs e)
        {
```

```
            try {//Stop the video stream
                stream.Stop();
            }catch(Exception ex)
            { MessageBox.Show(ex.Message); }
        }

        private void btnCONNECT_Click(object sender, EventArgs e)
        {
            try {//Select & connect the bluetooth device
                serialPort1.PortName =
this.cmbCOMP.GetItemText(this.cmbCOMP.SelectedItem);
                serialPort1.BaudRate = 9600;
                serialPort1.Open();
                if (!serialPort1.IsOpen) return;
                btnCONNECT.Enabled = false;

                MessageBox.Show("Connected", "Notification", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }catch(Exception ex)
            { MessageBox.Show(ex.Message); }
        }

        private void btnDISCONNECT_Click(object sender, EventArgs e)
        {
            try
            {//Disconnect the bluetooth
                if (serialPort1.IsOpen)
                serialPort1.Close();
            btnCONNECT.Enabled = true;
            MessageBox.Show("Disconnected", "Notification", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }

        }
        //Send Commands
        private void btnFORWARD_Click(object sender, EventArgs e)
        {
            try {
                serialPort1.Write("C");
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }

        }

        private void btnBACKWARD_Click(object sender, EventArgs e)
        {
            try
            {
                serialPort1.Write("B");
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }
        }

        private void btnLEFT_Click(object sender, EventArgs e)
        {
            try
            {
                serialPort1.Write("D");
```

```
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }
        }

        private void btnRIGHT_Click(object sender, EventArgs e)
        {
            try
            {
                serialPort1.Write("E");
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }
        }

        private void btnACTIVATEARM_Click(object sender, EventArgs e)
        {
            try { //select the text file
                string path = Environment.CurrentDirectory + "/" + "abc.txt";
                if (!File.Exists(path))
                {
                    MessageBox.Show("File not found");
                }
                else
                {
                    using (StreamReader sr = new StreamReader(path))
                    {//Read data from text file
                        string text = sr.ReadLine();
                        try
                        {
                            serialPort1.Write(text);
                        }
                        catch (Exception ex)
                        { MessageBox.Show(ex.Message); }
                    }


                }
            }
            catch (Exception ex)
            { MessageBox.Show(ex.Message); }

        }
    }
}
```

Image processing codes

Code 1 : Code for reading the colors of the background

```
url = 'http://172.16.101.96:8080/shot.jpg';
ri  = imread(url);
fh = image(ri);
z = 1;
i = 0;
s = zeros(240,320,'uint32');
t = zeros(240,320,'uint32');
u = zeros(240,320,'uint32');
while(z<=1)
    ri = imread(url);
    z = z+1;
    set(fh,'CData',ri);
    drawnow;
end

    for jj = 1:320
        for ii = 1:240
          x = ri(ii, jj,:);

          A = x(:,:,1);
          B = x(:,:,2);
          C = x(:,:,3);

          s(ii,jj) = A;
          t(ii,jj) = B;
          u(ii,jj) = C;

          %A
          %B
          %C
        end
    end


           a = unique(s);
           b = unique(t);
           c = unique(u);
%
 out1 = [a,histc(s(:),a)];
 out2 = [b,histc(t(:),b)];
 out3 = [c,histc(u(:),c)];
 %out
 fprintf('out1');
 out1
 fprintf('out2');
 out2
 fprintf('out3');
 out3
```

Code 2 : Codes for checking the color reduction of the surface versus object

```
url = 'http://172.16.101.96:8080/shot.jpg';
ri  = imread(url);
fh = image(ri);

while(1)
    ri = imread(url);
    mask = ((ri(:, :, 1) >= 92)&(ri(:, :, 1) <= 170))&((ri(:, :, 2) >=
71)&(ri(:, :, 2) <= 170)&((ri(:, :, 3)>=81)&(ri(:, :, 3)<=248)));
    maskedRgbImage = bsxfun(@times, ri, cast(mask, 'like', ri));
    set(fh,'CData',maskedRgbImage);
    drawnow;
end
```

Code 3 : Codes for extracting object out of the background

```
url = 'http://172.16.101.96:8080/shot.jpg';
ri  = imread(url);
fh = image(ri);
count = 0;
while(1)
    ri = imread(url);
    mask = ((ri(:, :, 1) >= 92)&(ri(:, :, 1) <= 170))&((ri(:, :, 2) >=
71)&(ri(:, :, 2) <= 170)&((ri(:, :, 3)>=81)&(ri(:, :, 3)<=248)));
    maskedRgbImage = bsxfun(@times, ri, cast(mask, 'like', ri));
    cropped = imcrop(maskedRgbImage,[160 120 10 10]);

    for jj = 1:10
        for ii = 1:10
          x = cropped(ii, jj,:);
          if (x(:,:,1)&x(:,:,2)&x(:,:,3)) == 0
           count = count + 1;
          end
          %disp(count);
          if(count>90)

              fprintf('Object Identified');
              count = 0;

%               word = 'A';
%               fileID = fopen('C:\Users\dev\Desktop\MATLAB.txt','w');
%               typed = fprintf(fileID,'%c\n',word);
%
              %jj = 200;

          else

%               word = '';
%               fileID = fopen('C:\Users\dev\Desktop\MATLAB.txt','w');
%               typed = fprintf(fileID,'%c\n',word);
```

```
            end
        end
    end
    set(fh,'CData',cropped);
    drawnow;
end
```

Method of working on the Image Processing Code.

- First Initiate the code 1 and run the code in Matlab.
- There will be 3 outputs called out1, out2, out3. In here, out1 is the Red color pixel values and no of red pixels contain in the image for each pixel value found in the image(there are actually 2 output values in out1,2,3), likewise out2 is for Green pixels and out3 gives the Blue pixels.
- Note down the ranges of pixels which included in the image regarding each RGB color which contains at least 1 pixel. It should be a range between 0 -255. Eg: out1 range will be 37-122 (which means pixels from 37-122 are present in the image), and there can be 100, 37-valued pixels and there will be no 36-valued pixels. Therefore 36 is not included in the range. (Once the code runs this will be clearly understood).
- Replace the ranges regarding out1, 2, 3 in the Code2. Eg: `((ri(:, :, 1) >= 37)&(ri(:, :, 1) <= 122)` according to the above example. Replace the ranges for all the outs.
- Now run the code2 and see whether the color reduction is occurring.
- If successful, then move to the code3 and replace the ranges with the same amounts as in code2.
- There will be a 10*10 pixels appear as the output. Keep an object in the middle of the camera image and the 10*10 pixel square should be blackened and output in the command prompt should be appeared as “object identified”.
- If unsuccessful switch off the lights to remove the shadows and try again. It should be successful. (Do not drag objects extremely fast to the camera sight. It can blackened the middle pixels and suddenly an object will be falsely identified.)

Future Works

- Development of ROS, AI algorithms necessary for proper decision taking on the image processed.
- Enhancement of the performance of the present robots with applied feedback controls to the motor and with better Kinematics models.
- Application of better Embedded System modules and Microcontrollers (Raspberry Pi).
- Production of commercially viable robot with all the necessary documents included.