Build Your Own Computer Controlled Robots from Scratch

Robotics for Processing IDE

Rajashekhar V S [1]

March 21, 2020

[1] http://vsrajashekhar.weebly.com/

Dedicated to my
country - India

Contents

List of Figures

Preface

Robotics is a fascinating field that is gaining importance in the recent decades. Robots have so far been controlled through a pre-programmed way and also through sensor based feedback. In this book, I have focused on controlling robots through a computer via a microprocessor. Moreover you can make these robots at your home.

There are six projects in this books that teaches you how to build various aspects of a robot. By doing these six projects, you will get confidence to build robotic projects that have graphical user interface with it.

It took nearly 1 year for me to conclude that Processing IDE can be used as a good graphical user interface for robots. Back in the year 2018, I started building my own robots using Processing IDE as my graphical user interface. I have build manipulators and mobile robots using it. In this book, I am sharing a few of my projects that seemed to be interesting for people around the world. I posted my videos of the robots with GUI on the YouTube and I received E-Mails from countries like China, France, Turkey and India. These mails were regarding the code that I have used for making the robot. Hence I have decided to write a book that would help people build their own computer controlled robots at their desktop.

My vision is make people build their own computer controlled robots at their home and use it in their day to day life. I hope my vision comes true by 2030!

-Rajashekhar V S

1

Introduction

Computer controlled robots are fascinating! Aren't they?

In this book we are going to explore about six projects that are controlled via a computer using a graphical user interface. The Chapter 2 gives information about the websites from where you can get more details about the Arduino and Processing IDE. The Chapter 3 gives you the links from where you can download and install the Arduino and Processing IDE software.

The Chapter 4 tells you how to control the individual LEDs and the RGB LED using a GUI. The GUI is shown in Figure 1.1. You would need 5 different LEDs and 1 RGB LED, a webcamera (or a built-in laptop camera), Arduino UNO and a few Male to Male jumper wires for this project.

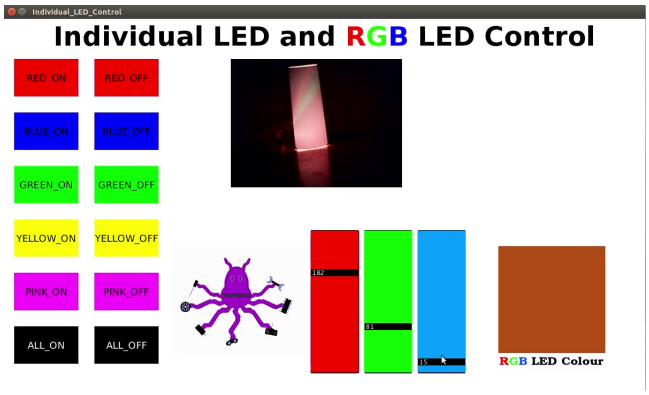


Figure 1.1: The buttons on the left are used to control the individual LEDs. The buttons in the mid bottom are used to control the colour of the RGB LED. The camera feedback is given on the mid top region.

The Chapter 5 teaches you how to build a foldable stair. The Figure 1.2 shows the GUI with the real foldable stair.

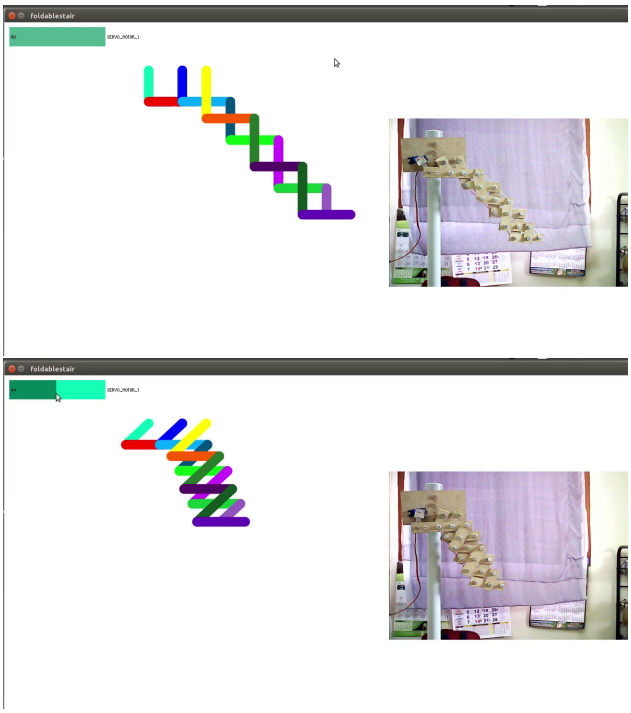


Figure 1.2: TOP: The foldable stair in the expanded position with the GUI; BOTTOM: The foldable stair in the contracted position with the GUI

In Chapter 6 you will learn how to build a 3 axis manipulator. It is as shown in Figure 1.3

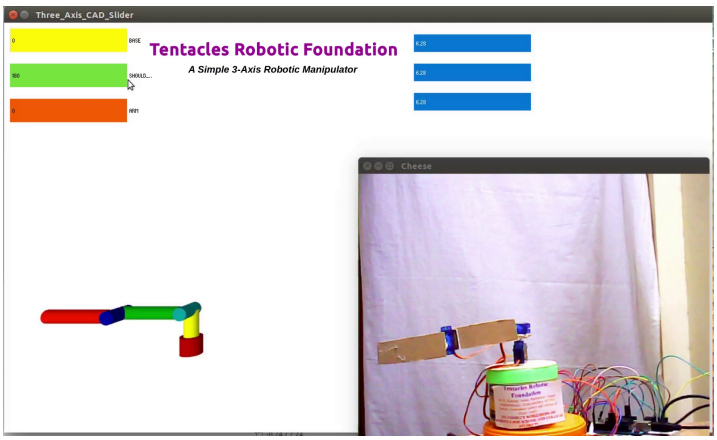


Figure 1.3: The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD robot at the bottom and physical robot moves accordingly.

In Chapter 7 you learn to create a five-bar mechanism using a graphical

user interface. The Figure 1.4 shows the expected output from the project.

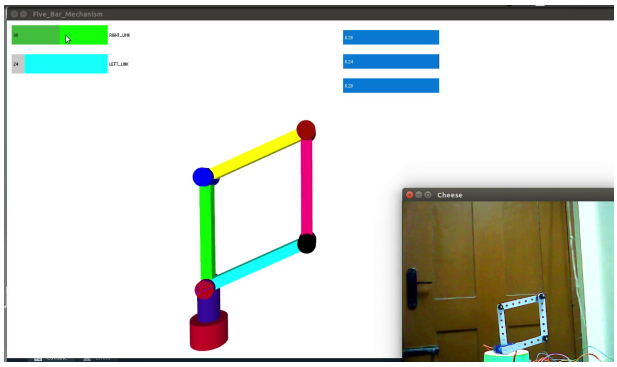


Figure 1.4: The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD model of the closed loop mechanism at the bottom and physical mechanism moves accordingly.

The Chapter 8 teaches you to simulate a GUI for a lizard. The expected output can be seen in the Figure 1.5.

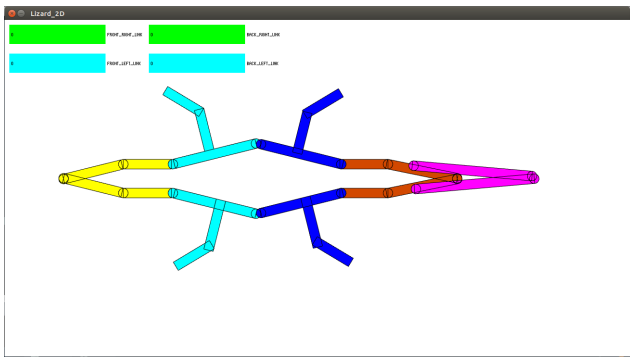


Figure 1.5: The top left of the figure contains the slider for adjusting the angle of the links. The CAD model of the closed loop mechanism at the bottom moves accordingly.

Image processing plays a vital role in robotics. Hence I have focused on it too. It can be seen in the Chapter 9. The trajectory of the ball has been traced in the Figure 1.6.

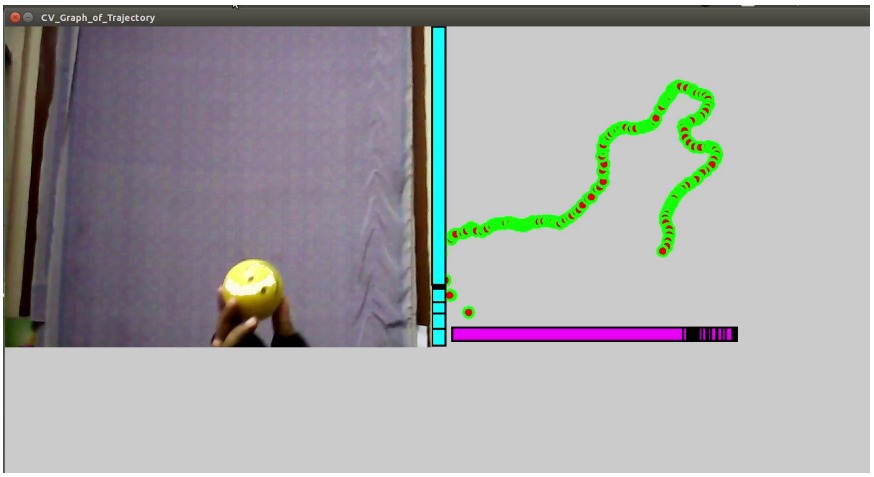


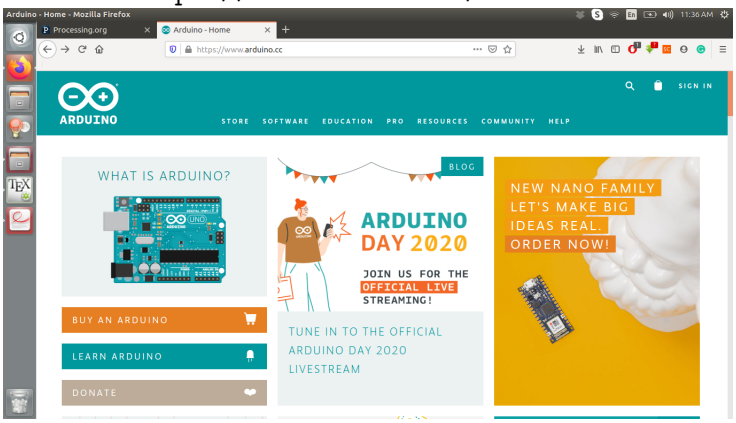
Figure 1.6: The top left of the figure contains the slider for adjusting the angle of the links. The CAD model of the closed loop mechanism at the bottom moves accordingly.

2

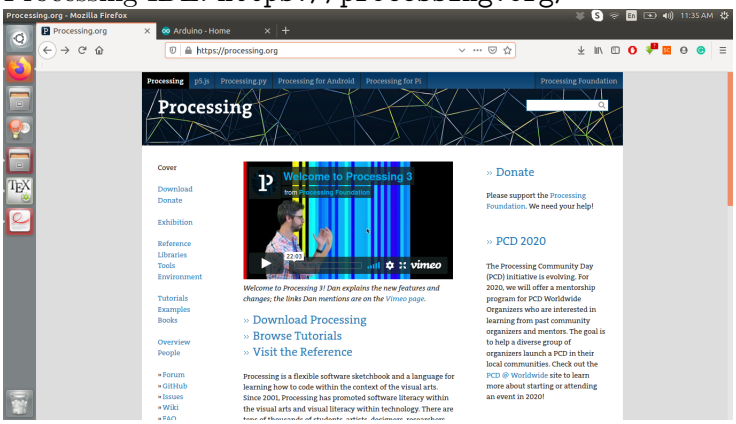
Arduino and Processing IDE

I do not want to waste much of your time here. You can refer to the following websites for more details about them.

Arduino: `https://www.arduino.cc/`



Processing IDE: `https://processing.org/`

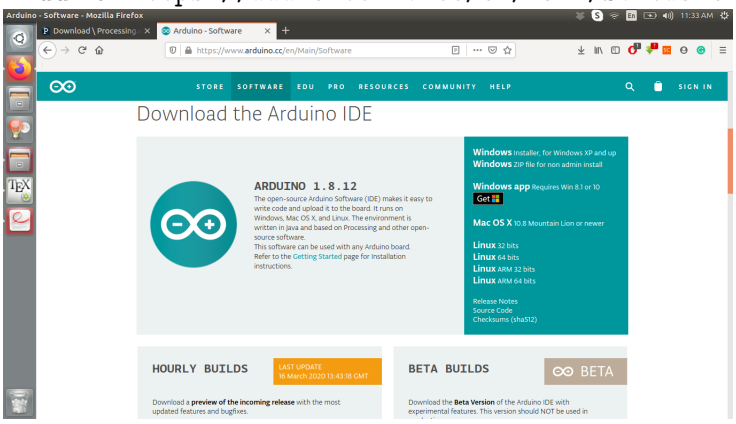


3

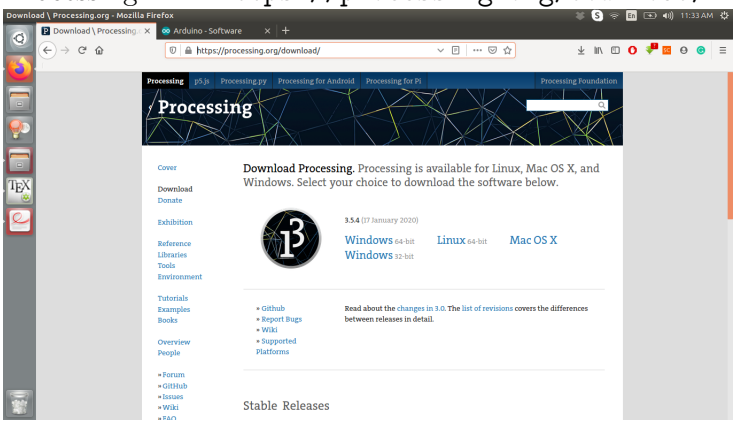
Installing the Arduino and Processing IDE

They can be downloaded and installed from here:

Arduino: `https://www.arduino.cc/en/Main/Software`



Processing IDE: `https://processing.org/download/`



4

A GUI for LED's and RGB LED's with Camera Feedback

LEDs are facinating when controlled through a computer. In this Chapter, we are going to see how to create a Graphical User Interface for single colour Light Emitting Diodes and RGB Light Emitting Diode. At the click of the mouse, the desired colour LED would glow.

The reader can take a quick look at the outcome of this chapter in this video: `https://youtu.be/3xd2Vv7tc4o`

4.1 Circuit Diagram

To create the circuit, you would need 5 different colour LED, in this case, red, blue, green, yellow, pink. You would also need a RGB LED, a bread board and an Arduino UNO with USB-UART cable. The Arduino UNO has to connected to the computer throughout the experiment. Now connect the circuit as in Figure 4.1.

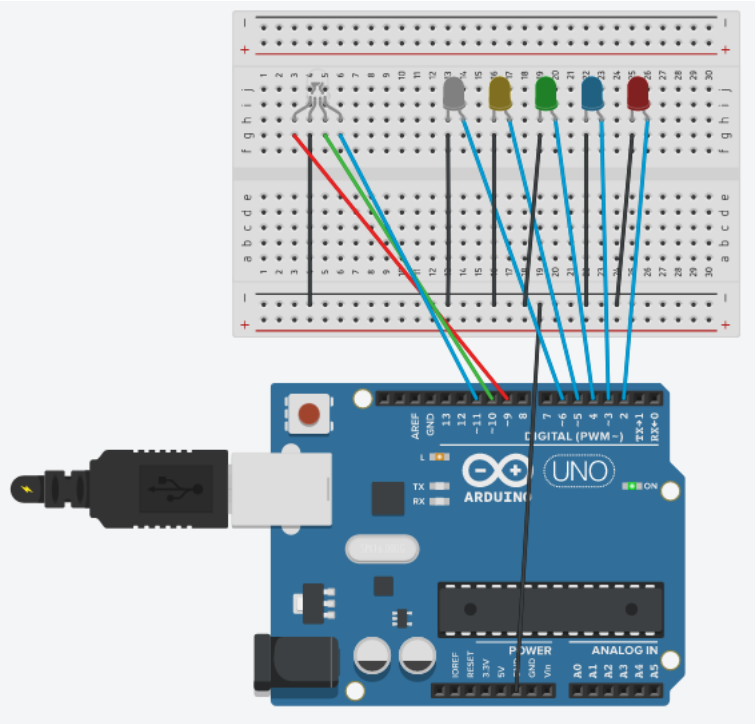


Figure 4.1: The circuit diagram of the 5 LEDs and 1 RGB LED connected to the Arduino UNO

4.2 Experimental Setup

In the experimental setup the 5 LEDs and 1 RGB LED are connected to the Arduino UNO. There is a webcamera infront of the LEDs that display the condition on the GUI screen. The experimental setup is as shown in Figure 4.2.

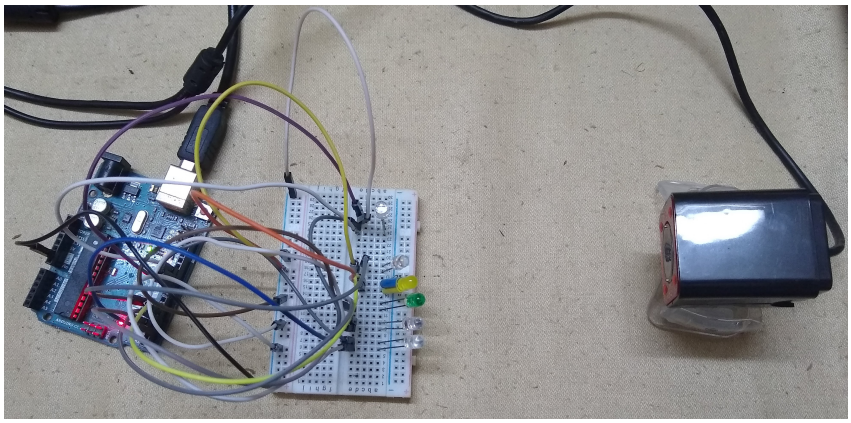


Figure 4.2: The experimental setup where 5 LEDs and 1 RGB LED are connected to the Arduino UNO. There is a webcamera infront of the LEDs that display the condition on the GUI

4.3 The Graphical User Interface

The Graphical User Interface (GUI) is used to interact with the LEDs and obtain the status on the screen through a camera. It can be seen in Figure 4.3. The buttons on the left are used to control the individual LEDs. The buttons in the mid bottom are used to control the colour of the RGB LED. The camera feedback is given on the mid top region.

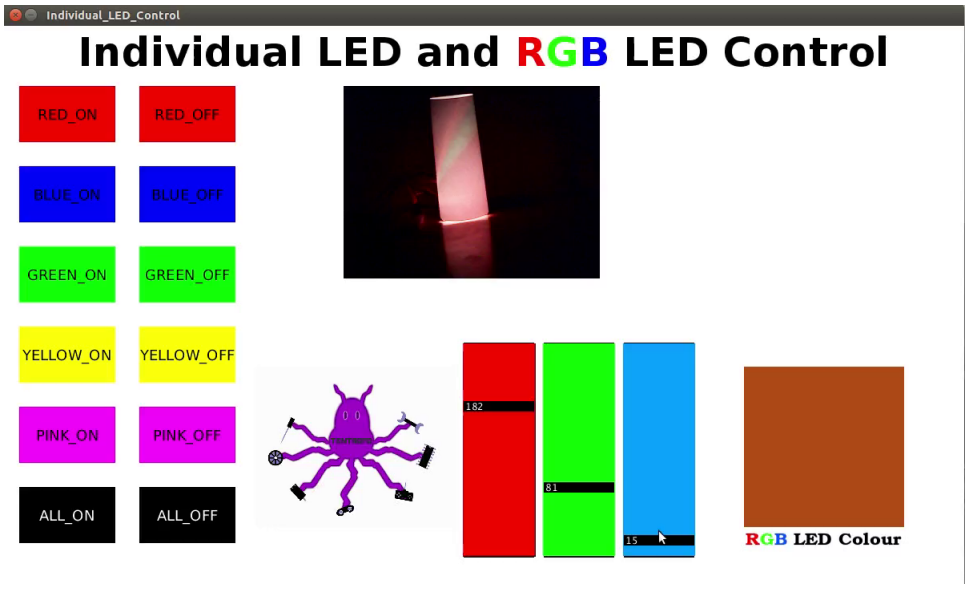


Figure 4.3: The buttons on the left are used to control the individual LEDs. The buttons in the mid bottom are used to control the colour of the RGB LED. The camera feedback is given on the mid top region.

4.4 Arduino code

This code is only for your reference. Download the code from here to use in your project: `https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/Individual%20LED%20and%20RGB%20LED%20Control.zip`

```
//code begins here
void setup() {
  //normal LEDs
  pinMode(2, OUTPUT);    //set pin as output , red led
  pinMode(3, OUTPUT);    //set pin as output , blue led
  pinMode(4, OUTPUT);    //set pin as output , green led
  pinMode(5, OUTPUT);    //set pin as output , yellow led
  pinMode(6, OUTPUT);    //set pin as output , pink led
    //RGB LEDs
    pinMode(8, OUTPUT); // ground
    pinMode(9, OUTPUT); // red
    pinMode(10, OUTPUT); // green
    pinMode(11, OUTPUT); // blue
  Serial.begin(9600);     //start serial communication @9600 bps
}
void loop() {
  if(Serial.available()){  //id data is available to read
```

```
    char val = Serial.read();
//red
    if(val == 'r'){        //if r received
      digitalWrite(2, HIGH); //turn on red led
      }
    if(val == 'a'){        //if a received
      digitalWrite(2, LOW); //turn off red led
      }
//blue
    if(val == 'b'){        //if b received
      digitalWrite(3, HIGH); //turn on blue led
      }

    if(val == 'c'){        //if c received
      digitalWrite(3, LOW); //turn off blue led
      }
//green
    if(val == 'g'){        //if g received
      digitalWrite(4, HIGH); //turn on green led
      }
    if(val == 'd'){        //if d received
      digitalWrite(4, LOW); //turn off green led
      }
//yellow
    if(val == 'y'){        //if y received
      digitalWrite(5, HIGH); //turn on yellow led
      }
    if(val == 'e'){        //if e received
      digitalWrite(5, LOW); //turn off yellow led
      }
//pink
    if(val == 'p'){        //if p received
      digitalWrite(6, HIGH); //turn on pink led
      }
    if(val == 'f'){        //if f received
      digitalWrite(6, LOW); //turn off pink led
      }
////////////////////////////////////////////
//////////FULL INDIVIDUAL ON/OFF CONTROL
////////////////////////////////////////s///
//all individual off
    if(val == 'x'){        //if x received
      digitalWrite(2, LOW); //turn off all led
      digitalWrite(3, LOW);
```

```
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
      analogWrite(9, 0);
      analogWrite(10, 0);
      analogWrite(11, 0);
      }
//all individual on
    if(val == 'z'){        //if z received
      digitalWrite(2, HIGH); //turn on all led
      digitalWrite(3, HIGH);
      digitalWrite(4, HIGH);
      digitalWrite(5, HIGH);
      digitalWrite(6, HIGH);
//      analogWrite(9, 255);
//      analogWrite(10,255);
//      analogWrite(11,255);
      }
////////////////////////////////
/////////RGB Individual Control
////////////////////////////////
    if(val == 'j'){        //if p received
analogWrite(9, GetFromSerial());
      }
    if(val == 'k'){        //if f received
analogWrite(10, GetFromSerial());
      }
    if(val == 'l'){        //if f received
analogWrite(11, GetFromSerial());
      }
  }
}
// read the serial port
int GetFromSerial()
{
while (Serial.available()<=0) {
  }
return Serial.read();
}
//code ends here
```

4.5 Processing IDE code

This code is only for your reference. Download the code from here to use in your project: https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/Individual%20LED%20and%20RGB%20LED%20Control.zip

```
//code begins here
import controlP5.*; //import ControlP5 library
import processing.serial.*;
import processing.video.*;
Serial port;

ControlP5 cp5; //create ControlP5 object
sliderV sV1, sV2, sV3;

PFont zigBlack;
Capture cam;
PImage img;
PImage logo;
PImage rgb;

void setup(){

size(1200, 700);     //window size, (width, height)
background(255);
PFont font = createFont("Georgia",12);
//camera starts
        String[] cameras = Capture.list();

         if (cameras.length == 0) {
        println("There are no cameras available for capture.");
         exit();
        } else {
        println("Available cameras:");
        for (int i = 0; i < cameras.length; i++) {
        println(cameras[i]);
        }

        // The camera can be initialized directly using an
        // element from the array returned by list():
        cam = new Capture(this, cameras[0]);
        cam.start();
         }
 //camera over
```

```
  printArray(Serial.list());
   //prints all available serial ports

  port = new Serial(this, "/dev/ttyACM0", 9600);
  //i have connected arduino to /dev/ttyACM0, it would be
  //different in windows and mac os

  cp5 = new ControlP5(this);
         zigBlack = createFont("Serif.bolditalic-48.vlw", 20);
         font = createFont("CenturySchL-BoldItal-48", 18);
// custom fonts for buttons and title

  cp5.addButton("Red_ON")
    .setPosition(20, 75)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,0,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,0,0))

  ;
  cp5.addButton("Red_OFF")
    .setPosition(170, 75)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,0,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,0,0))

  ;

  cp5.addButton("Blue_ON")
    .setPosition(20, 175)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(0,0,255))
```

```
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,0,255))
  ;

  cp5.addButton("Blue_OFF")
     .setPosition(170, 175)
     .setSize(120, 70)
     .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(0,0,255))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,0,255))
  ;

  cp5.addButton("Green_ON")
     .setPosition(20, 275)
     .setSize(120, 70)
     .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(0,255,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,255,0))
  ;

     cp5.addButton("Green_OFF")
     .setPosition(170, 275)
     .setSize(120, 70)
     .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(0,255,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,255,0))
  ;

     cp5.addButton("Yellow_ON")
     .setPosition(20, 375)
     .setSize(120, 70)
     .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,255,0))
```

```
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,255,0))
  ;

  cp5.addButton("Yellow_OFF")
    .setPosition(170, 375)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,255,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,255,0))
  ;

   cp5.addButton("Pink_ON")
    .setPosition(20, 475)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,0,255))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,0,255))
  ;

     cp5.addButton("Pink_OFF")
    .setPosition(170, 475)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(255,0,255))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(255,0,255))
  ;

   cp5.addButton("All_ON")
    .setPosition(20, 575)
    .setSize(120, 70)
    .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(255)
                .setColorActive(color(0,0,0))
```

```
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,0,0))
  ;

    cp5.addButton("All_OFF")
   .setPosition(170, 575)
   .setSize(120, 70)
   .setFont(font)
                .setColorValue(color(0))
                .setColorLabel(255)
                .setColorActive(color(0,0,0))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,0,0))
  ;

 //RGB Slider Creation starts

  sV1 = new sliderV(575, 400, 90, 255, #FF0000);
  sV2 = new sliderV(675, 400, 90, 255, #03FF00);
  sV3 = new sliderV(775, 400, 90, 255, #009BFF);

  img = loadImage("robot.png");
  logo = loadImage("logo.jpg");
  rgb = loadImage("rgbledcolour.png");
}

void draw(){

  sV1.render();
  sV2.render();
  sV3.render();

  fill(sV1.p,sV2.p,sV3.p);
  rect(925, 425, 200, 200);

// send sync character
// send the desired value
  port.write('j');
  port.write(sV1.p);
  port.write('k');
  port.write(sV2.p);
  port.write('l');
  port.write(sV3.p);
```

```
  if (cam.available() == true) {
    cam.read();
  }
  image(cam, 425, 75,320,240);
  image(img, 0, 5, 1200, 50);
  //image(logo, 900, 425, 250, 200);
  image(logo, 315, 425, 250, 200);
  image(rgb, 925, 630, 200, 20);

}


void Red_ON(){
  port.write('r');
}

void Red_OFF(){
  port.write('a');
}

void Blue_ON(){
  port.write('b');
}

void Blue_OFF(){
  port.write('c');
}

void Green_ON(){
  port.write('g');
}

void Green_OFF(){
  port.write('d');
}

void Yellow_ON(){
  port.write('y');
}

void Yellow_OFF(){
  port.write('e');
}
```

```
void Pink_ON(){
  port.write('p');
}

void Pink_OFF(){
  port.write('f');
}

void All_ON(){
  port.write('z');
}

void All_OFF(){
  port.write('x');
}


class sliderV {
int x, y, w, h, p;
  color cor;
boolean slide;

  sliderV (int _x, int _y, int _w, int _h, color _cor) {
    x = _x;
    y = _y;
    w = _w;
    h = _h;
    p = 90;
    cor = _cor;
    slide = true;
  }

void render() {
    fill(cor);
    rect(x-1, y-4, w, h+10);
    noStroke();
    fill(0);
    rect(x, h-p+y-5, w-2, 13);
    fill(255);
    text(p, x+2, h-p+y+6);

    if (slide==true && mousePressed==true && mouseX<x+w && mouseX>x){
      if ((mouseY<=y+h+150) && (mouseY>=y-150)) {
        p = h-(mouseY-y);
```

```
if (p<0) {
            p=0;
          }
else if (p>h) {
            p=h;
          }
        }
      }
    }
}
//code ends here
```

5

A Simple Foldable Stair with GUI

In regions where space is a constrain, foldable stairs can be used. In this chapter we are going to build a GUI for the foldable stairs. It will be useful to know the folded position of the stairs if this GUI is used. This serves like a feedback although there are no sensors used.

A video of the outcome of what is going to be done in this chapter can be seen here: `https://youtu.be/c4dZuiiyhz0`

5.1 Circuit Diagram

The project involves the usage of a micro servo motor and an Arduino UNO with USB-UART cable. The circuit is simple and can be seen in Figure 5.1.

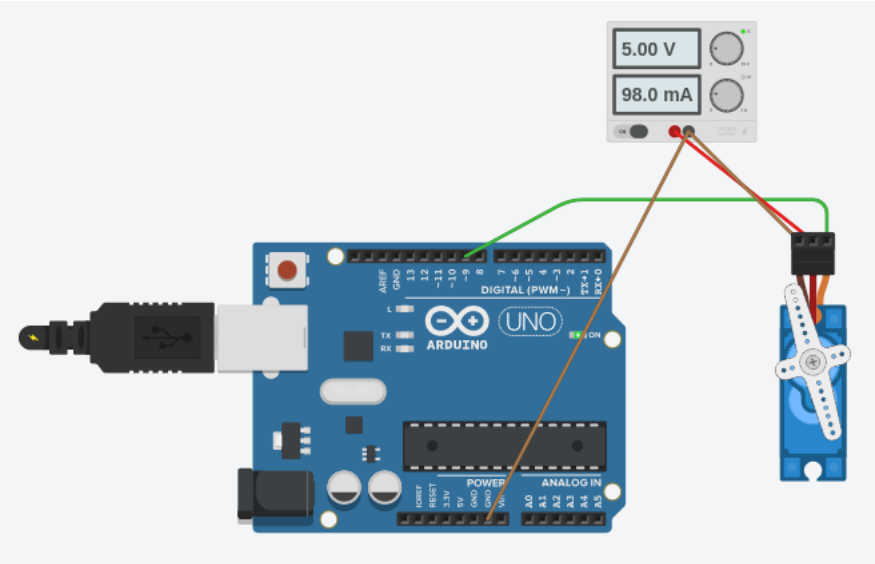


Figure 5.1: The circuit diagram of the servo motor connected to the Arduino UNO

5.2 Experimental Setup

The mechanism involved can be seen in this paper [1]. You can make it out by looking into the Figure 5.2. The prototype of the stair can be made using cardboard as linkages[2] and board pins as joints[3].

[1]Rajashekhar, V. S., K. Thiruppathi, and R. Senthil. "Modelling, Simulation and Control of a Foldable Stair Mechanism with a Linear Actuation Technique." Procedia Engineering 97 (2014): 1312-1321.

[2]The one which makes up the mechanism

[3]The one which connects two links

Figure 5.2: TOP: The prototype of the foldable staircase mechanism; BOTTOM: The prototype with the Arduino UNO connected to it

5.3 The Graphical User Interface

The graphical user interface can be created using the Processing IDE. It has provision to display the video captured in the camera. The Figure 5.3 shows the two positions of the stairs, this is one in the extended position and another in the folded position.

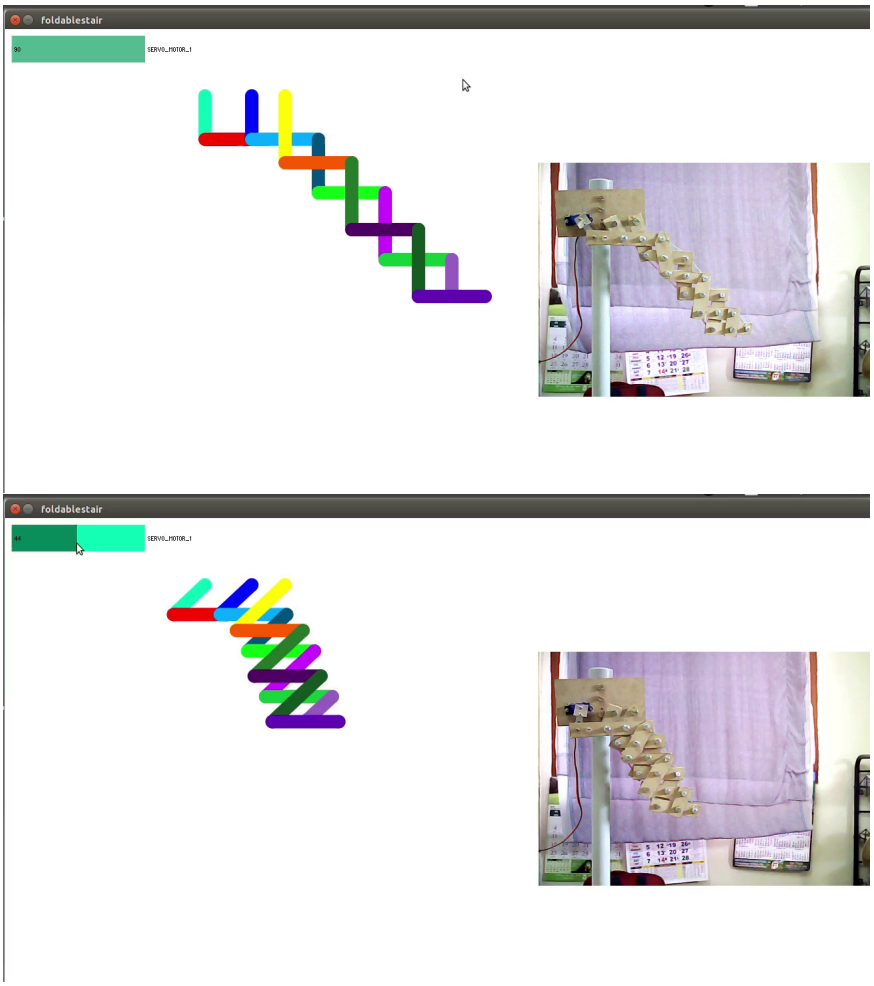


Figure 5.3: TOP: The foldable stair in the expanded position with the GUI; BOTTOM: The foldable stair in the contracted position with the GUI

5.4 Arduino code

Using the Arduino software, upload the StandardFirmata example (located in Examples > Firmata > StandardFirmata) to your Arduino board.

5.5 Processing IDE code

This code is only for your reference. Download the code from here to use in your project:`https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/staircase.zip`

```
import processing.serial.*;
```

```
import cc.arduino.*;
import controlP5.*;

import processing.video.*;
Capture cam;

ControlP5 controlP5;
Arduino arduino;
int Servo_Motor_1 = 0;

PImage img;

void setup()
{
  size(1300,700);
  smooth();
  frameRate(30);

  String[] cameras = Capture.list();

  if (cameras.length == 0) {
    println("There are no cameras available for capture.");
    exit();
  } else {
    println("Available cameras:");
    for (int i = 0; i < cameras.length; i++) {
      println(cameras[i]);
    }
    cam = new Capture(this, cameras[0]);
    cam.start();
  }


  arduino = new Arduino(this, "/dev/ttyACM0", 57600);
for (int i = 0; i <= 13; i++)
arduino.pinMode(i, Arduino.OUTPUT);
controlP5 = new ControlP5(this);
    controlP5.addSlider("Servo_Motor_1")
     .setPosition(10,10)
     .setSize(200,40)
     .setRange(0,90)
                  .setColorValue(color(0))
               .setColorLabel(0)
               .setColorActive(color(1,129,92))
```

```
                .setColorForeground(color(79,178,147))
                .setColorBackground(color(0,255,180))
                .setValue(90)
                //.setNumberOfTickMarks(5)
      ;

}
void draw()
{
  background(255);
  pushMatrix();
  translate(300, 100);
  drawLinkOne();
  drawLinkThree();
  popMatrix();
  arduino.analogWrite(8, (Servo_Motor_1+40));
  delay(0);
}


void drawLinkOne()//green
{
      if (cam.available() == true) {
    cam.read();
  }
  image(cam, 500, 100,500,350);

  //firstlink
  noStroke();
  fill(0, 255, 180);//pale green
  pushMatrix();
  translate(0,0);
  rotate(radians(90));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,60);
  ellipse(0, 0, 20, 20);
  ellipse(0, 60, 20, 20);
  drawLinkTwo();
  popMatrix();
}

  void drawLinkTwo()//red
{
  //secondlink
```

```
 pushMatrix();
  noStroke();
  fill(255, 0, 0);//red
  translate(0,65);
  rotate(radians(-180));
   rotate(radians(Servo_Motor_1));
  rect(-10,0,20,80);
  ellipse(0, 0, 20, 20);
  ellipse(0, 80, 20, 20);
  popMatrix();
}

 void drawLinkThree()//blue

 {
  //thirdlink
   pushMatrix();
  noStroke();
  fill(0, 0, 255);//blue
  translate(70,0);
  rotate(radians(90));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,60);
  ellipse(0, 0, 20, 20);
  ellipse(0, 60, 20, 20);
  drawstepone();
    popMatrix();


      //fourthlink
   pushMatrix();
  noStroke();
  fill(255,255,0);//yellow
  translate(120,0);
  rotate(radians(90));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstepthree();
    popMatrix();

}
```

```
void drawstepone()
{
          //fifthlink

  fill(0,168,255);//skyblue
   pushMatrix();
  translate(0,65);
  rotate(radians(180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawsteptwo();
    popMatrix();
}

void drawsteptwo()
{
          //fifthlink
     fill(2, 82 ,124);//grey blue
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,80);
  ellipse(0, 0, 20, 20);
  ellipse(0, 80, 20, 20);
  drawstepfourone();
    popMatrix();
}

void drawstepthree()
{
           fill(255,94,0);//florezent orange
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstepthreetwo();
    popMatrix();
}
```

```
void drawstepthreetwo()
{
        fill(35,116,40);//dark grey green
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstepfiveone();
    popMatrix();
}

void drawstepfourone()
{
fill(3,255,17);//florezent green
   pushMatrix();
  translate(0,80);
  rotate(radians(-180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstepfivetwo();
    popMatrix();
}

void drawstepfiveone()
{
  fill(84,0,103);//dark magenta
   pushMatrix();
  translate(0,100);
  rotate(radians(-180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
    drawstepsixone();
    popMatrix();
}

void drawstepfivetwo()
```

```
{
fill(207,0,255);//magenta
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstepsixtwo();
    popMatrix();
}

void drawstepsixone()
{
  fill(18,85,33);//green
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstep7one();
    popMatrix();
}

void drawstepsixtwo()
{
fill(15,191,54);//lighterthanabove
   pushMatrix();
  translate(0,100);
  rotate(radians(-180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
  drawstep7two();
    popMatrix();
}


void drawstep7one()
{
```

```
  fill(100,0,183);//pinkishblue
     pushMatrix();
  translate(0,100);
  rotate(radians(-180));
  rotate(radians(Servo_Motor_1));
  rect(-10,0,20,100);
  ellipse(0, 0, 20, 20);
  ellipse(0, 100, 20, 20);
    popMatrix();
}

void drawstep7two()
{
fill(151,97,196);//lighterthanabove
   pushMatrix();
  translate(0,100);
  rotate(radians(180));
  rotate(radians(-Servo_Motor_1));
  rect(-10,0,20,50);
  ellipse(0, 0, 20, 20);
  ellipse(0, 50, 20, 20);
    popMatrix();
}
```

6

A 3 Axis Robotic Manipulator with GUI

A manipulator is inspired by the human arm. It can be used for various applications like pick and place, welding, etc. Creating a simple 3 axis manipulator with a graphical user interface is the objective of this chapter.

One can see the following video to get an idea about what the output of this chapter would be: `https://youtu.be/fZdbQB4zLcU`

6.1 Circuit Diagram

An external power supply of 5V and about 1amps is required to power the three servos. The pulse width modulation signals are given from the Arduino UNO. One can use a breadboard to ease the circuit connections. Refer to Figure 6.1 and give the connections.

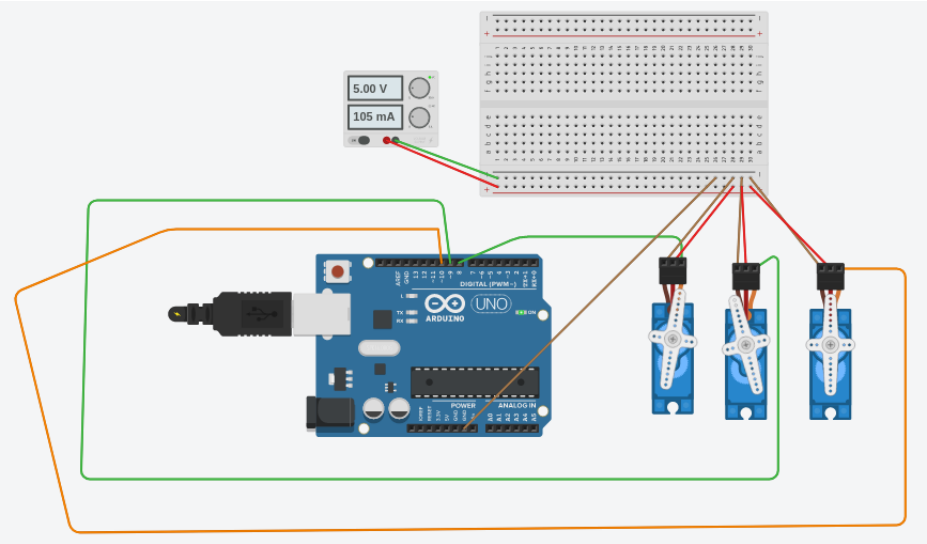


Figure 6.1: The circuit diagram of the 3 servo motors connected to the Arduino UNO through the power supply

6.2 Experimental Setup

The prototype of the simple 3 axis manipulator can be made using thick cardboard. One can use the RoboAnalyzer[1] software or read the book "Introduction to Robotics"[2] to understand more about manipulators. You can refer to Figure 6.2 for details about the physical model of the manipulator.

[1] www.roboanalyzer.com
[2] Saha, Subir Kumar. Introduction to robotics. Tata McGraw-Hill Education, 2014.

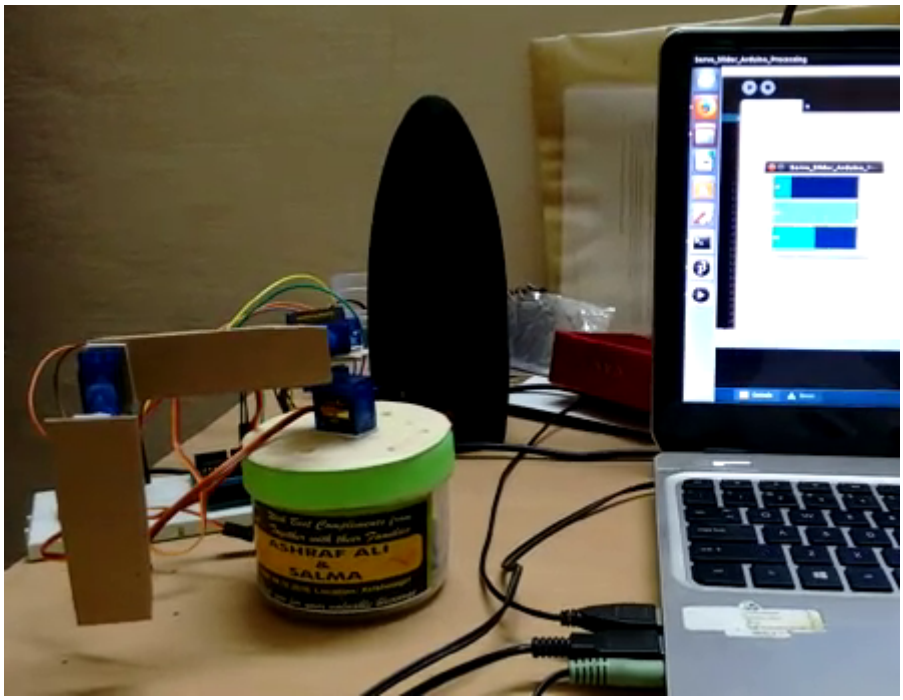


Figure 6.2: The 3 axis manipulator can be seen on the left side. Three 9g servos are connected in series.

6.3 The Graphical User Interface

The GUI created using the Processing IDE is shown in Figure 6.3. The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD robot at the bottom and physical robot moves accordingly. The camera feature is not in-built in this GUI. Hence you need to use an external application to view what is happening in reality.

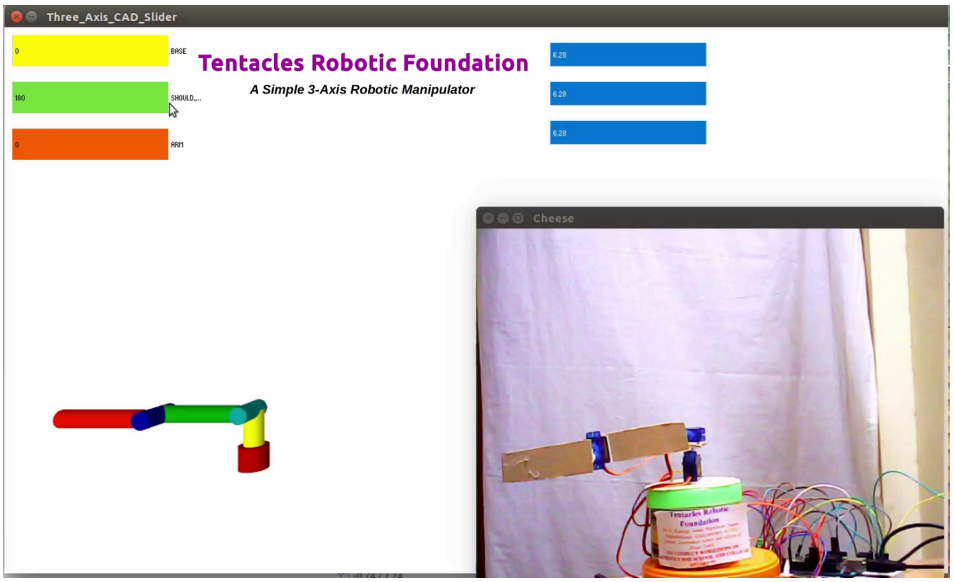


Figure 6.3: The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD robot at the bottom and physical robot moves accordingly.

6.4 Arduino code

Using the Arduino software, upload the StandardFirmata example (located in Examples > Firmata > StandardFirmata) to your Arduino board.

6.5 Processing IDE code

This code is only for your reference. Download the code from here to use in your project:`https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/three_axis_robot.zip`

```
import processing.serial.*;
import cc.arduino.*;
import controlP5.*;

float rotX, rotY, transX, transY, scaleFactor;

float offset = PI/24.0;

PShape base, body,shouldercylinder,arm;

ControlP5 controlP5;
ControlP5 MyController;
```

```
Arduino arduino;

int Base = 0;
int Shoulder= 0;
int Arm= 0;

void setup()

{
size(1300,700, P3D);

  smooth();
  frameRate(30);

  MyController = new ControlP5(this);

  MyController.addSlider("X_Rotation",0,6.28,20,700,20,200,30);
  MyController.addSlider("Y_Rotation",0,6.28,20,700,70,200,30);
  MyController.addSlider("Z_Rotation",0,6.28,20,700,120,200,30);

body = loadShape("body.obj");
base = loadShape("base.obj");
shouldercylinder = loadShape("shouldercylinder.obj");
arm = loadShape("arm.obj");

  arduino = new Arduino(this, "/dev/ttyACM0", 57600);
for (int i = 0; i <= 13; i++)
arduino.pinMode(i, Arduino.OUTPUT);
controlP5 = new ControlP5(this);
    controlP5.addSlider("Base")
     .setPosition(10,10)
     .setSize(200,40)
     .setRange(0,180)
                .setColorValue(color(0))
                .setColorLabel(0)
              .setColorActive(color(219,217,60))
                .setColorForeground(color(200,200,200))
                  .setColorBackground(color(255, 255, 0))

     ;

    controlP5.addSlider("Shoulder")
     .setPosition(10,70)
     .setSize(200,40)
```

```
      .setRange(0,180)
                      .setColorValue(color(0))
                .setColorLabel(0)
              .setColorActive(color(113,211,58))
                .setColorForeground(color(200,200,200))
                  .setColorBackground(color(0, 255, 0))
      ;

     controlP5.addSlider("Arm")
      .setPosition(10,130)
      .setSize(200,40)
      .setRange(0,180)
                              .setColorValue(color(0))
                .setColorLabel(0)
              .setColorActive(color(252,149,84))
                .setColorForeground(color(200,200,200))
                  .setColorBackground(color(255,100, 0))
      ;
}


float X_Rotation = 0;
float Y_Rotation = 0;
float Z_Rotation = 0;


void draw()
{
lights();

background(255,255,255);
  pushMatrix();
  translate(320, 550);
  rotateY(X_Rotation);
  rotateX(Y_Rotation);
  rotateZ(Z_Rotation);
  drawBaseFixed();
  popMatrix();

  arduino.analogWrite(8, Base);
  arduino.analogWrite(9, Shoulder);
  arduino.analogWrite(10, Arm);
  delay(0);
}
```

```
void drawBaseFixed()
{

  drawLinkBase();
}


  void drawLinkBase()
{
  pushMatrix();
    translate(0,0,0);
        scale(0.12);
      shape(base);
  popMatrix();

      pushMatrix();

      rotateY(radians(Base));
      translate(0,0,0);
      scale(0.12);
      shape(body);

      translate(0,-500,0);
      rotateZ(radians(-Shoulder)+radians(-90));
      shape(shouldercylinder);

      translate(0,825,200);
      rotateZ(radians(-Arm));
      shape(arm);

      popMatrix();
}
```

7

A Closed Loop Mechanism with GUI

A closed loop mechanism occurs commonly in the field of robotics. For example, four-bar mechanism, five-bar mechanism are all closed loop mechanisms. In this chapter, we will see how to create a GUI for a five-bar mechanism and control it through a computer.

You can see the expected output of this chapter in the following video: `https://youtu.be/MKp2IFxrDDM`

7.1 Circuit Diagram

An external power supply of 5V and about 1amps is required to power the two servos. The pulse width modulation signals are given from the Arduino UNO. One can use a breadboard to ease the circuit connections. Refer to Figure 7.1 and give the connections.

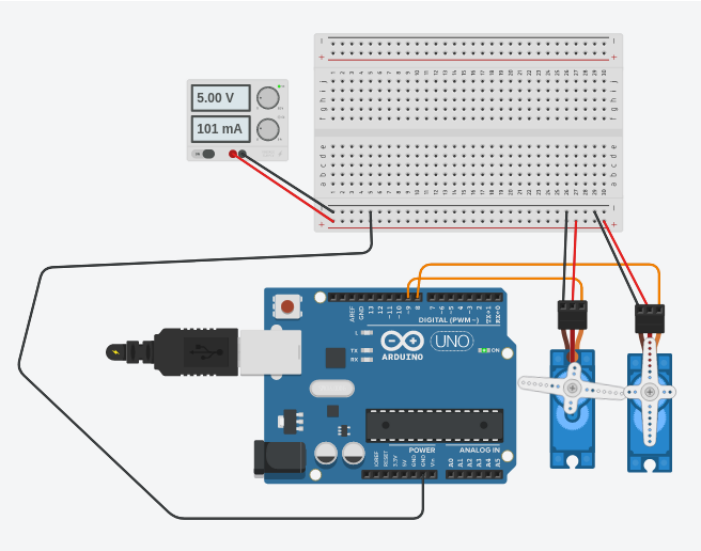


Figure 7.1: The circuit diagram of the 2 servo motors connected to the Arduino UNO through the power supply

7.2 Experimental Setup

Refer to the Figure 7.2. You can use linkages from the builder set or thick cardboard to build the linkages. The joints can be made of board pins. There are two motors place one behind the other in the Figure. From the motor shaft, the linkages are connected.

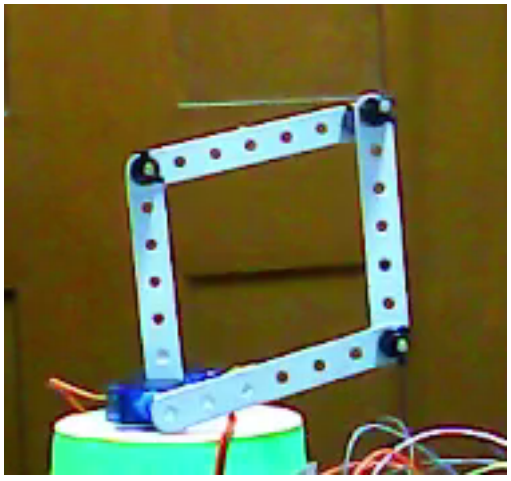


Figure 7.2: The closed loop five bar mechanism can be seen in this figure. Two 9g servos are connected in parallel at the bottom.

7.3 The Graphical User Interface

The GUI created using the Processing IDE is shown in Figure 7.3. The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD mechanism at the bottom and physical mechanism moves accordingly. The camera feature is not in-built in this GUI. Hence you need to use an external application to view what is happening in reality.

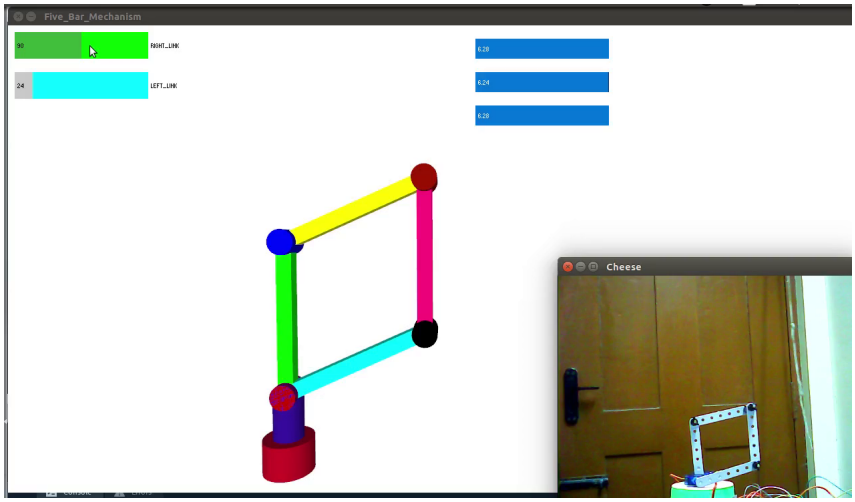


Figure 7.3: The top left of the figure contains the slider for adjusting the angle of the servo motors. The CAD model of the closed loop mechanism at the bottom and physical mechanism moves accordingly.

7.4 Arduino code

Using the Arduino software, upload the StandardFirmata example (located in Examples > Firmata > StandardFirmata) to your Arduino board.

7.5 Processing IDE code

This code is only for your reference. Download the code from here to use in your project:`https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/closed_loop_mechanism.zip`

```
import processing.serial.*;
import cc.arduino.*;
import controlP5.*;

float rotX, rotY, transX, transY, scaleFactor;
```

```
float offset = PI/24.0;

PShape L1, L2,L3,L4,base;

ControlP5 controlP5;
ControlP5 MyController;
Arduino arduino;

int right_link = 90;
int left_link= 0;

void setup()

{

size(1300,700, P3D);

  smooth();
  frameRate(30);

  MyController = new ControlP5(this);

  MyController.addSlider("X_Rotation",0,6.28,20,700,20,200,30);
  MyController.addSlider("Y_Rotation",0,6.28,20,700,70,200,30);
  MyController.addSlider("Z_Rotation",0,6.28,20,700,120,200,30);

        L1 = loadShape("linkone.obj");
        L2= loadShape("linktwo.obj");
        L3= loadShape("linkthree.obj");
        L4= loadShape("linkfour.obj");
        base =loadShape("base.obj");

  arduino = new Arduino(this, "/dev/ttyACM0", 57600);
  for (int i = 0; i <= 13; i++)
  arduino.pinMode(i, Arduino.OUTPUT);
  controlP5 = new ControlP5(this);

              controlP5.addSlider("right_link")
               .setPosition(10,10)
               .setSize(200,40)
               .setRange(0,180)
               .setColorValue(color(0))
               .setColorLabel(0)
               .setColorActive(color(57,173,56))
```

```
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,255,0))
                ;

              controlP5.addSlider("left_link")
                .setPosition(10,70)
                .setSize(200,40)
                .setRange(0,180)
                .setColorValue(color(0))
                .setColorLabel(0)
                .setColorActive(color(73,175,175))
                .setColorForeground(color(200,200,200))
                .setColorBackground(color(0,255,255))
                ;

}

float X_Rotation = 0;
float Y_Rotation = 0;
float Z_Rotation = 0;

void draw()
{

lights();
background(255,255,255);

                pushMatrix();
              translate(420, 550);
              rotateY(X_Rotation);
              rotateX(Y_Rotation);
              rotateZ(Z_Rotation);
              drawBaseFixed();
                popMatrix();

 arduino.analogWrite(8, right_link);
 arduino.analogWrite(9, left_link);
  delay(0);

}

void drawBaseFixed()
{
  drawLinkBase();
```

```
}

  void drawLinkBase()
{
              pushMatrix();
           translate(0,110,0);
           rotateX(radians(90));
           scale(0.75);
           shape(base);
              popMatrix();

             pushMatrix();
         rotateX(radians(-90));
         rotateY(radians(right_link));
         translate(0,0,0);
         scale(0.75);
         shape(L1);
         translate(300,0,0);
         rotateY(radians(-90));
         rotateY(radians(left_link)+radians(0));
         rotateY(radians(-right_link)+radians(90));
         shape(L3);
             popMatrix();

              pushMatrix();
         translate(0,0,0);
         rotateX(radians(270));
         rotateY(radians(left_link));
         scale(0.75);
         shape(L2);
         translate(300,0,0);
         rotateY(radians(90));
         rotateY(radians(-left_link));
         rotateY(radians(right_link)+radians(-90));
         shape(L4);
              popMatrix();
}
```

8

A Robotic Lizard with GUI

A robotic lizard is created in this chapter which would replicate a real lizard. The movements are simulated. There is no hardware implementation of this. However the reader can try to implement it in hardware in co-ordination with the GUI.

You can see the expected output of this chapter in the following video: `https://youtu.be/D5PpKxY8nzg`

8.1 The Graphical User Interface

The top of the Figure 8.1 contains the slider to move the various parts of the lizard. The bottom of it contains the lizard which moves according to the commands of the user.

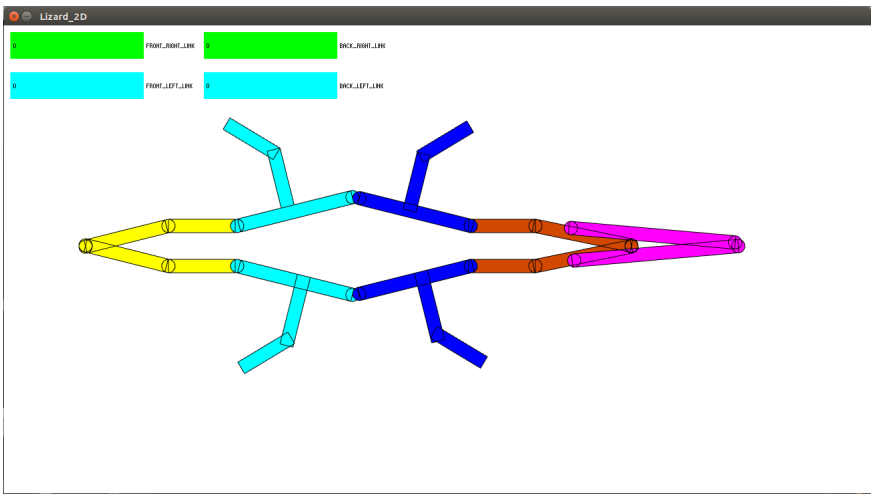


Figure 8.1: The top left of the figure contains the slider for adjusting the angle of the links. The CAD model of the closed loop mechanism at the bottom moves accordingly.

8.2 Processing IDE code

This code is only for your reference. Download the code from here to use in your project:https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/Lizard_2D.zip

```
import processing.serial.*;
import controlP5.*;

float rotX, rotY, transX, transY, scaleFactor;

float offset = PI/24.0;

PShape L1, L2,L3,L4,base;

ControlP5 controlP5;
int front_right_link = 0;
int front_left_link= 0;
int back_right_link = 0;
int back_left_link= 0;

void setup()
{

size(1300,700, P3D);

  smooth();
  frameRate(30);
for (int i = 0; i <= 13; i++)
  controlP5 = new ControlP5(this);

                controlP5.addSlider("front_right_link")
                 .setPosition(10,10)
                 .setSize(200,40)
                 .setRange(0,50)
                 .setColorValue(color(0))
                 .setColorLabel(0)
                 .setColorActive(color(57,173,56))
                 .setColorForeground(color(200,200,200))
                 .setColorBackground(color(0,255,0))
                 ;

                controlP5.addSlider("front_left_link")
                 .setPosition(10,70)
```

```
                    .setSize(200,40)
                    .setRange(0,50)
                    .setColorValue(color(0))
                    .setColorLabel(0)
                    .setColorActive(color(73,175,175))
                    .setColorForeground(color(200,200,200))
                    .setColorBackground(color(0,255,255))
                    ;

                     controlP5.addSlider("back_right_link")
                    .setPosition(300,10)
                    .setSize(200,40)
                    .setRange(0,50)
                    .setColorValue(color(0))
                    .setColorLabel(0)
                    .setColorActive(color(57,173,56))
                    .setColorForeground(color(200,200,200))
                    .setColorBackground(color(0,255,0))
                    ;

                   controlP5.addSlider("back_left_link")
                    .setPosition(300,70)
                    .setSize(200,40)
                    .setRange(0,50)
                    .setColorValue(color(0))
                    .setColorLabel(0)
                    .setColorActive(color(73,175,175))
                    .setColorForeground(color(200,200,200))
                    .setColorBackground(color(0,255,255))
                    ;

}

void draw()
{
lights();
background(255,255,255);
                  pushMatrix();
                translate(700, 300);
                drawBaseFixed();
                  popMatrix();
  delay(0);
}
```

```
void drawBaseFixed()
{
  //tail starts here
  pushMatrix();
        rotateZ(radians(-back_right_link));
        fill(210,72,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,100,20);
        translate(100,10,0);
        ellipse(0,0,20,20);

        rotateZ(radians((back_left_link)*0.625));
        rotateZ(radians((back_right_link)));
        rotateZ(radians(11.5));
        fill(210,72,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,150,20);
        translate(150,10,0);
        ellipse(0,0,20,20);

  popMatrix();

    pushMatrix();
        translate(0,60,0);
        rotateZ(radians(back_left_link));
        fill(210,72,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,100,20);
        translate(100,10,0);
        ellipse(0,0,20,20);

        rotateZ(radians((-back_right_link)*0.625));
        rotateZ(radians(-back_left_link));
        rotateZ(radians(-11.5));
        fill(210,72,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,150,20);
        translate(150,10,0);
        ellipse(0,0,20,20);
  popMatrix();
```

```
  //final tail begins
  pushMatrix();
        rotateZ(radians((-back_right_link)/2));
        rotateZ(radians((back_left_link)/2));
        rotateZ(radians(5));
        translate(150,-10,0);
        fill(255,0,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,250,20);
        translate(250,10,0);
        ellipse(0,0,20,20);
  popMatrix();

    pushMatrix();
        rotateZ(radians((back_left_link)/2));
        rotateZ(radians((-back_right_link)/2));
        rotateZ(radians(-5));
        translate(150,65,0);
        fill(255,0,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,250,20);
        translate(250,10,0);
        ellipse(0,0,20,20);
    popMatrix();
  //tail ends here

  //head starts here
pushMatrix();
    translate(-350,0,0);
    pushMatrix();
        rotateZ(radians(front_right_link));
        fill(255,255,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-100,20);
        translate(-100,10,0);
        ellipse(0,0,20,20);

        rotateZ(radians((-front_left_link)*0.75));
        rotateZ(radians((-front_right_link)));
        rotateZ(radians(-14));
```

```
        fill(255,255,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-125,20);
        translate(-125,10,0);
        ellipse(0,0,20,20);
  popMatrix();

    pushMatrix();
        translate(0,60,0);
        rotateZ(radians(-front_left_link));
        fill(255,255,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-100,20);
        translate(-100,10,0);
        ellipse(0,0,20,20);

        rotateZ(radians((front_right_link)*0.75));
        rotateZ(radians(front_left_link));
        rotateZ(radians(14));
        fill(255,255,0);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-125,20);
        translate(-125,10,0);
        ellipse(0,0,20,20);
    popMatrix();
popMatrix();
  //head ends here

  //body begins here
  //left body
  pushMatrix();
        translate(-350,0,0);
        rotateZ(radians((front_right_link)/2));
        rotateZ(radians((back_right_link)/2));
        rotateZ(radians(166));
        fill(0,255,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-175,20);
        translate(-175,10,0);
        ellipse(0,0,20,20);
```

```
        translate(87.5,10,0);
        rotateZ(radians(-90));
        rect(0,0,-87.5,20);
        translate(-87.5,0,0);
        rotateZ(radians(-45));
        rect(0,0,-87.5,20);
  popMatrix();

    pushMatrix();
        translate(-350,60,0);
        rotateZ(radians((-back_left_link)/2));
        rotateZ(radians((-front_left_link)/2));
        rotateZ(radians(194));
        fill(0,255,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,-175,20);
        translate(-175,10,0);
        ellipse(0,0,20,20);
        translate(87.5,10,0);
        rotateZ(radians(90));
        rect(0,0,-107.5,20);
        translate(-87.5,7,0);
        rotateZ(radians(45));
        rect(0,0,-87.5,20);
  popMatrix();

  //right body
    pushMatrix();
        rotateZ(radians((-back_right_link/2)));
        rotateZ(radians(-front_right_link/2));
        rotateZ(radians(194));
        fill(0,0,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,175,20);
        translate(175,10,0);
        ellipse(0,0,20,20);
        translate(-87.5,0,0);
        rotateZ(radians(-90));
        rect(0,0,-87.5,20);
        translate(-75,7,0);
        rotateZ(radians(45));
        rect(0,0,-87.5,20);
```

```
    popMatrix();

    pushMatrix();
        translate(0,60,0);
        rotateZ(radians((front_left_link/2)));
        rotateZ(radians(back_left_link/2));
        rotateZ(radians(166));
        fill(0,0,255);
        ellipse(0,0,20,20);
        translate(-3,-10,0);
        rect(0,0,175,20);
        translate(175,10,0);
        ellipse(0,0,20,20);
        translate(-87.5,10,0);
        rotateZ(radians(90));
        rect(0,0,-107.5,20);
        translate(-100,0,0);
        rotateZ(radians(-45));
        rect(0,0,-87.5,20);
     popMatrix();
}
```

9

Image Processing: 2D Trajectory Tracking of a Ball Using a Camera

Image processing is a crucial part of robotics. In this chapter we will see how to trace the trajectory of an object that moves infront of the camera.

You can see the expected output of this chapter in the following video: `https://youtu.be/qcTFj0wEqlM`

9.1 Experimental Setup

The experiment requires a camera (webcamera) to be connected to the computer. The build-in camera of the laptop will also work fine.

9.2 The Graphical User Interface

Once the Processing code is loaded, the camera is activated. Show an object of unique colour that is not present in the background. Now click on the colour and start moving the object slowly. You will find the object tracing a path on the right side as shown in Figure 9.1.

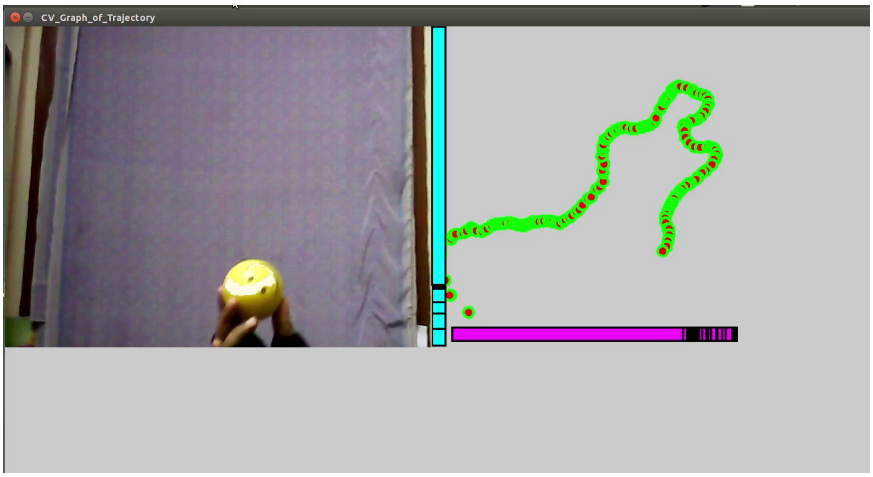


Figure 9.1: The top left of the figure contains the slider for adjusting the angle of the links. The CAD model of the closed loop mechanism at the bottom moves accordingly.

9.3 Processing IDE code

This code is only for your reference. Download the code from here to use in your project:`https://github.com/rajashekharvs91/Computer-Controlled-Robots/blob/master/CV_Graph_of_Trajectory.zip`

```
import processing.serial.*;
import cc.arduino.*;
import processing.video.*;
import controlP5.*;
Capture video;
Arduino arduino;
int Base=90;

color trackColor;
float threshold = 25;

void setup() {
  size(1366, 768);
  smooth();
  //frameRate(30);
  String[] cameras = Capture.list();
  printArray(cameras);
  video = new Capture(this, cameras[3]);
  video.start();
  trackColor = color(255, 255, 0);
```

```
}
void captureEvent(Capture video)
{
  video.read();
}
void draw() {
  trackvideo();
    delay(0);
}
void trackvideo()
{
  video.loadPixels();
  image(video, 0, 0);
  //threshold = map(mouseX, 0, width, 0, 100);
  threshold = 80;
  float avgX = 0;
  float avgY = 0;
  int count = 0;
  // Begin loop to walk through every pixel
  for (int x = 0; x < video.width; x++ ) {
    for (int y = 0; y < video.height; y++ ) {
      int loc = x + y * video.width;
      // What is current color
      color currentColor = video.pixels[loc];
      float r1 = red(currentColor);
      float g1 = green(currentColor);
      float b1 = blue(currentColor);
      float r2 = red(trackColor);
      float g2 = green(trackColor);
      float b2 = blue(trackColor);
      float d = distSq(r1, g1, b1, r2, g2, b2);
      if (d < threshold*threshold) {
        stroke(255);
        strokeWeight(1);
        point(x, y);
        avgX += x;
        avgY += y;
        count++;
      }
    }
  }
 if (count > 0)
 {
    avgX = (avgX) / (1*count);
```

```
        avgY = (avgY) / (1*count);
        // Draw a circle at the tracked pixel
        fill(255,0,0);
        strokeWeight(4);
        stroke(0,255,0);
        translate(600,0);
        ellipse(avgX, avgY, 24, 24);
        //x-axis
        fill(255,0,255);
        strokeWeight(4);
        stroke(0,0,0);
        translate(50,500);
        rect(0,0,avgX-30,20);
        //y-axis
        fill(0,255,255);
        strokeWeight(4);
        stroke(0,0,0);
        translate(0,-450);
        rect(0,0,-20,avgY);
      }
    }
float distSq(float x1,float y1,float z1,float x2,float y2,float z2) {
      float d = (x2-x1)*(x2-x1) + (y2-y1)*(y2-y1) +(z2-z1)*(z2-z1);
      return d;
    }
    void mousePressed() {
      // Save color where the mouse is clicked in trackColor variable
      int loc = mouseX + mouseY*video.width;
      trackColor = video.pixels[loc];
    }
```

10

Conclusion

In this book we have done 6 projects using Arduino UNO and Processing IDE. The first project taught you how to turn on and off LED lights of different colours. It also taught you how to control the RGB LED using the Processing IDE. The second project taught you to control a single servo motor. This inturn was used to control the foldable stairs. The third project taught you to control a 3 axis robotic manipulator. You can explore by adding additional joints and grippers to this. The fourth project taught you how to create a GUI for closed loop mechanisms and control the same. The fifth project was simulation based and a robotic lizard was simulated. You can build your own hardware and then control the same. In the sixth and the last project, you were taught how to track a particular object and plot it by the side. By understanding these six projects and doing them on your own, I think you must be capable enough to work on more challenging projects like snake robots, scorpion robots and so on.

The reader can now explore various other options that are available by combining two or three of these projects together. You can also make new projects with the concepts learnt from this book. The readers are welcome to give their feedback, suggestions and comments on this book to the author Rajashekhar V S by contacting him via his
E-Mail ID: vsrajashekhar@gmail.com