ANIMATRONICS FACE: PROF HOOD

Mario Navarro
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
nmario597@gmail.com

Monique Lopez
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
moniquej-lopez@hotmail.com

Victor Palos
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
victorpalosT@gmail.com

Submitted as part of a project for ME4543 Mechatronics, Fall 2018

TABLE OF CONTENTS

ABSTRACT

This report details a project using an Arduino Uno to create a rudimentary mechatronics system by making an animatronic face of Professor Lyle Hood of the UTSA Mechanical Engineering Department. The frame of the system was constructed using fiberboard and wood and held together with epoxy and screws. An ultrasonic sensor and thermistor were used and interfaced with three servo motors which acted as the eyes and mouth. Once a basic version of the electronics were made, the electronics and frame were interfaced and perfected into the final product shown in the video linked in this report.

Section 1: Literature review

Design 1:

One design found was one that featured moving eyebrows, eyes, and jaw. The eyebrows are made out of short plastic tubings that were wrapped in black electrical tape. These eyebrows were then screwed on directly to 2 servo motors. The brows have three states, angled upwards, angled downward, and angled horizontally. The face was created out of a thin round piece of medium density fiberboard. This mount resembles a puppet type mouth, where the entire lower jaw moves. The jaw moves by one servo motor attached to the edge. Two holes where he eyes are to be placed were cut out of the face material. The eyes work by creating a gimbal type design for the eye structure. Both the vertical and horizontal axises are attached to two separate servo motors for both eyes. In total, this robot has 7 servo motors. This project does have sound as a function of the animatronic, but that feature was not explained in the tutorial.

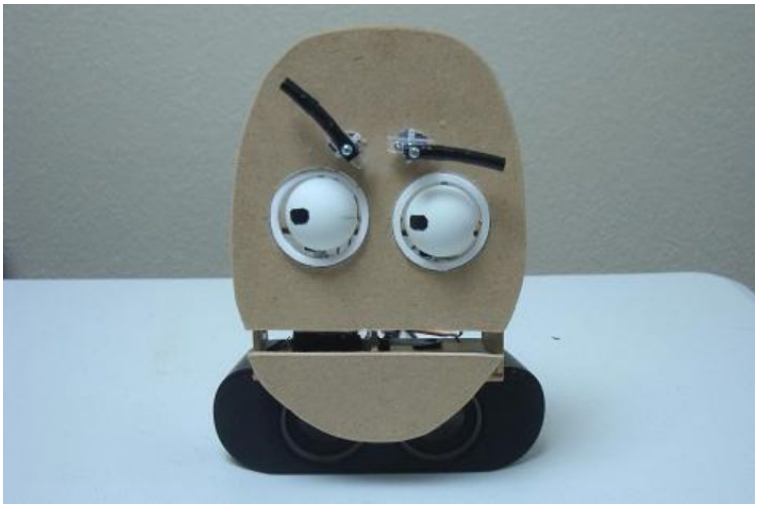

Figure 1. Pyroelectric Animatronic

Design 2:

A second design used a coconut as the framework for the face. This would be ideal as it would save on money for something like an erector or lego and require less time assemble, but unfortunately the framework would have a shelf life. A coconut base would certainly expire between the times of presenting it for milestone 3 or 4 and for the actual demonstration on the November 30th. A second one could be made, presumably. The actual moving parts involved moving eyes and a light-up nose. However, the design used only one type of sensor in the form of 3 ultrasonic sensors. The design used the sensors to detect hand movement and triggered servos to move the eyes in the direction the hand. The nose lighted up whenever the system was triggered to demonstrate that the face was operable in the dark. The project was done by a hobbyist and seemed comparatively simple to the scale of the project required by this class. As a fair warning for the link, the final product is a bit unsettling.

Figure 2. CoconutHead Robot

Design 3:

Another design found was one that contained the features of moving the eyelids, jaw, and nose. For this design, they used a ball socket method over a 2-axis gimble. The ball socket would all for the addition of eyelids. The only issue would be that the gimble would require more powerful motors. For the ball socket, the receiver for the ball was created using a soda bottle. One section of the bottle was cut and wrapped around a glass marble. The heat was then used to form the plastic around the marble. A frame was then created using an erector set which could easily be replaced by either wood or legos. Four motors would be used to produce movement in the eyes. Each eye contains two motors for movement in the x and y-axis. The motors would be servo motors. The jaw was created using the erector set. The jaw was secured to the erector set using zip ties, this created a hinge-like form. A large servo motor would then be attached to the upper jaw, while the servo horn would be attached to the lower jaw. The servo motor would then be adjusted to 90 degrees. Once the jaw was finished the last moving piece would be the nose. Another motor would then be needed to the nose movement. The motor would be adjusted as the previous ones, as well being zip-tied and hot glued onto the erector set for security. Audio would be added to this face by saving the audio onto an SD card which is then placed into an MP3 shield which is attached to an Arduino mega. The link for this animatronic face is listed below.

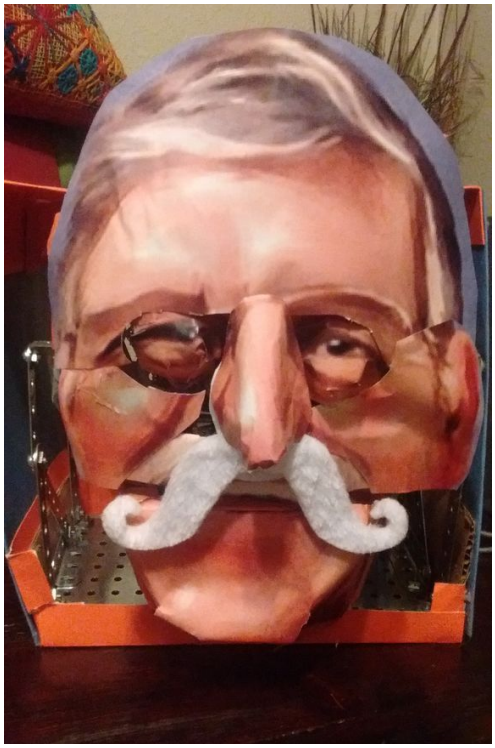

Figure 3. Romobot

Section 2: Brainstorming (initial planning)

We would like the Hood Bot to talk about memes. The face will be made out of cardboard and will have Dr. Hood's face pasted to the material. To meet the minimum requirements, at least the eyes and jaw would move through the use of servos. One of the sensor integrations would be an ultrasonic sensor to detect the proximity of anyone nearby and trigger the Hood bot to say a greeting or ask who made the plastic shrimp in Numerical Methods. It will be randomized and he may also give you extra credit. The second sensor integration would consist of a temperature sensor, where when it feels a warmer temperature it will react and output a comment. The Arduino and components will be connected/soldered directly to the frame and powered by someone's laptop or charging block. The soundtrack can be cleaned up and organized in an audio software such as Audacity and both the files and components can be edited and timed together.

Section 3: Supporting structure

The face was created by first printing out a full page sized picture of Dr. Hood. Once the image was printed, it was placed on top of a piece of fiberboard and sketched onto the board in order to cut the shape of the head. A piece of 2x4 was cut in order to create the chassis of the animatronic. The pieces of 2x4 were epoxied together as well as being attached with L-brackets. This same technique was used to attach the chassis to the base.

Section 4: Joints and motors

The joints created for this week are that of the jaw as well as those of the eyebrows. The simplest joint would be the eyebrows. Each eyebrow will contain a servo motor, which will give the eyebrow movement when one of the group members approaches hoodbot as well as when

he senses a change in temperature. As for the mouth, it will contain two servo motors to give it movement. The jaw will move whenever either the ultrasonic sensor or the thermistor is activated giving it a line of speech. The reasons servo motors were picked were due to it being readily available as well as financially inexpensive. The servos allow for sixty degrees of freedom, which would allow the eyebrows to rotate from zero to forty-five degrees. As for the jaw, it will contain two servo motors at the sides. The range for this servo will range from 10-20 degrees depending on the hinge that connects them to the face of hoodbot. You can see the wiring of the servo motors to the Arduino as well as the thermistor and the ultrasonic sensor to that of the breadboard in figure 4.

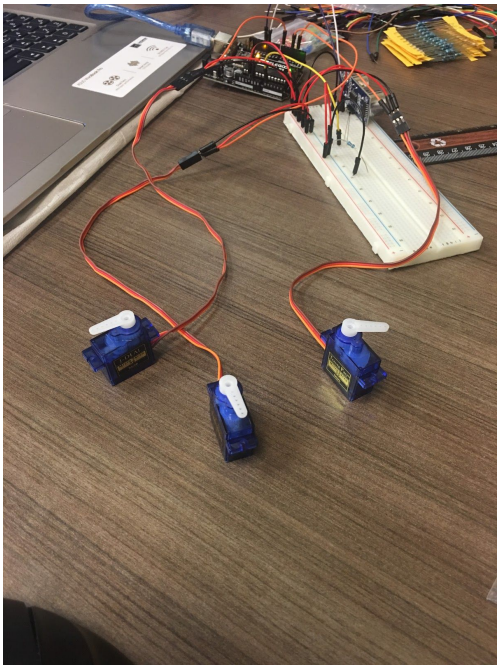

Figure 4. *Wiring*

Section 5: Sensors

The hoodbot will consist of two sensors, one being an ultrasonic sensor and the other being a thermistor. The ultrasonic sensor works by emitting sound waves at a high frequency that we as humans can not hear. The sensor then waits for sound to be reflected back, which it then calculates the distance based on the time it takes for the sound to be reflected back. The ultrasonic sensor is used on the hoodbot as a way to detect one of the group members at which hoodbot would then activate and state a code of line such as “Hey Victor” or any other group member. As for the thermistor, it is a variable resistor that changes their resistance depending on the temperature. In this case, as temperature increases the resistance in the thermistor then decreases. For the hoodbot when the thermistor senses an increase in temperature above 75 degrees Fahrenheit the servo motors would then activate and produce movement in both the eyebrows and jaw. It will also state a code of line along the lines of “it's getting toasty in here”. The thermistor and ultrasonic sensor can be seen at a closer view down below in figure 5.

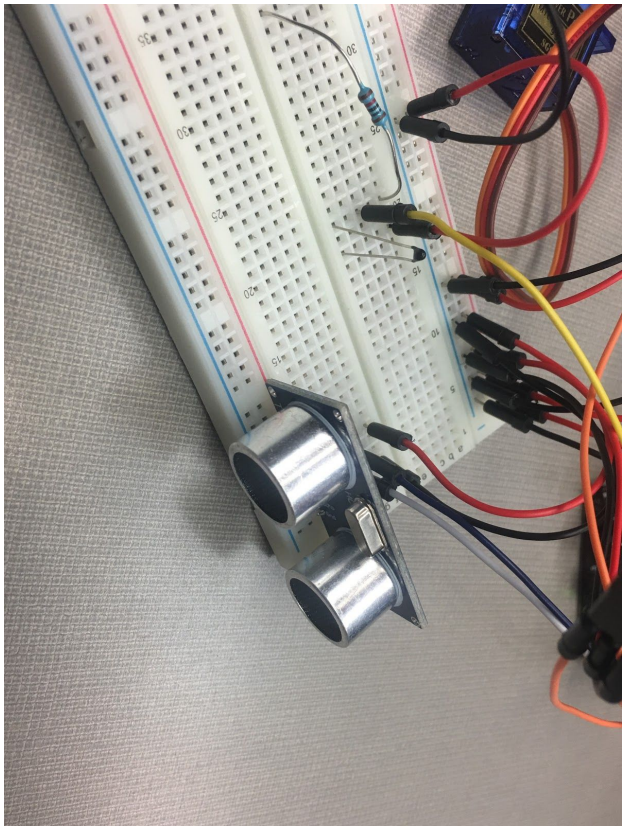

Figure 5: Thermistor and Ultrasonic Sensor

Section 6: Programming for interaction

The initialization and setup function for this code set all the variables for the three servo motors, ultrasonic sensor, thermistor, and mp3 module. The ultrasonic sensor and thermistor, when triggered, activated two separate reaction utilizing delays to keep the triggers from overlapping each other. For the ultrasonic trigger, when an object was detected within 10in of the sensor, the hoodbot would raise his eyebrows with two servo motors, speak with a third servo motor, then repeat the process synced with the mp3 module. For the thermistor trigger, when touched and made to detect a temperature above 78°F, caused hoodbot to speak with his eyebrows and mouth together in sync with a different audio file. To sync the motors, a series of for loops and delays were used to define the motor movement.

YouTube Link: https://youtu.be/iXWE2J5ibDo

Section 7: Lessons learned and suggestions

1. Learning how to code more in detail with Arduino and being able to diagnose where the problem is within the code.
2. How to integrate audio into Arduino.
3. Sometimes you'll have faulty components, and it's recommended having extra components for when those break down.
4. Keep the lab the same, the milestones really helped keep the project on schedule.
5. Have more available time slots for the lab as well as have multiple days.

Section 8: Personnel and bill of materials
(a) Personnel

Task	Main Personnel	Secondary personnel
Chassis Design	Monique	Victor, Mario
Construction Of Chassis	Mario and Victor	Monique
Sensor Integration Coding	Victor and Monique	Mario
Servo Timing	Mario	Victor, Monique
Report Writing	Victor, Mario, Monique	

(b) Bill of materials

No.	Description	Website/comment	Qty.	Unit $	Total $
1	Arduino UNO	Already owned by a group member	1	free	free
2	Servo Motors	Amazon	5	11.78	11.78
3	Wood & Fiberboard	Homedepot	1	18.36	18.36
4	Speakers	Amazon	1	14.99	14.99
5	MP3 Module	Amazon	1	7.99	7.99
6	Micro SD Card	Target	1	7.57	7.57
7	Jigsaw	Walmart	1	28.86	28.86
	Total	Total cost of the project		89.55	89.55

Acknowledgments
We would like to thank Dr. Hood for being the dankest professor and providing us with great content for our robot.

References:

[1] Animatronics How To DIY
http://www.pyroelectro.com/animatronics-how-to/

[2] ROMOBOT - ANIMATRONIC FACE ROBOT
https://www.instructables.com/id/RomoBOT-Animatronic-Face-Robot/

[3] CoconutHead Arduino Based Animatronic Project
https://www.youtube.com/watch?v=CT9iG5XxCSw

Appendix A: Code

```
// Servo Initialization
#include <Servo.h>
Servo eyeL;
Servo eyeR;
Servo jaw;
int angleL;
int angleR;
int angleJ;

// Test Variables
Servo servo1;
Servo servo2;
int angle;

// Ultrasonic Initialization
const int pingPin = 12; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 11; // Echo Pin of Ultrasonic Sensor

// Thermistor Initialization
int ThermistorPin = 0;
int Vo;
float R1 = 10000;
float logR2, R2, T;
float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 = 2.019202697e-07;

// Audio
#include <SoftwareSerial.h>

#define ARDUINO_RX 5//should connect to TX of the Serial MP3 Player module
#define ARDUINO_TX 6//connect to RX of the module
SoftwareSerial mySerial(ARDUINO_RX, ARDUINO_TX);

static int8_t Send_buf[8] = {0} ;
#define CMD_NEXT_SONG     0X01  // Play next song
#define CMD_VOLUME_UP     0X04
#define CMD_PLAY_W_INDEX 0X03
#define CMD_SET_VOLUME 0X06
#define CMD_SEL_DEV 0X09
#define DEV_TF 0X02
#define CMD_PLAY 0X0D
#define CMD_PAUSE 0X0E
#define CMD_SINGLE_CYCLE 0X19
#define SINGLE_CYCLE_ON 0X00
```

```
#define SINGLE_CYCLE_OFF 0X01
#define CMD_PLAY_W_VOL 0X22

void setup() {
  // Servo Setup
  servo1.attach(7);
  servo2.attach(4);
  jaw.attach(2);
  servo1.write(angleL);
  servo2.write(angleR);
  jaw.write(angleJ);

  // Ultrasonic/Thermistor Setup
  Serial.begin(9600);

  // Audio Set Up
  mySerial.begin(9600);
  delay(500);//Wait chip initialization is complete
    sendCommand(CMD_SEL_DEV, DEV_TF);//select the TF card
  delay(200);//wait for 200ms
}

long microsecondsToInches(long microseconds) {
   return microseconds / 74 / 2;
}

void loop() {
  // Ultrasonic Trigger
   long duration, inches;
   pinMode(pingPin, OUTPUT);
   digitalWrite(pingPin, LOW);
   delayMicroseconds(2);
   digitalWrite(pingPin, HIGH);
   delayMicroseconds(10);
   digitalWrite(pingPin, LOW);
   pinMode(echoPin, INPUT);
   duration = pulseIn(echoPin, HIGH);
   inches = microsecondsToInches(duration);

  if (inches <= 10){
    angleL = 60;
    angleR = 120;
    servo1.write(angleL);
    servo2.write(angleR);
    sendCommand(CMD_PLAY_W_VOL, 0X1E02);
```

```
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
    delay(20);
   }
   angleL = 60;
   angleR = 120;
   servo1.write(angleL);
   servo2.write(angleR);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
    delay(20);
   }
   angleL = 60;
   angleR = 120;
   servo1.write(angleL);
   servo2.write(angleR);
    for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
    delay(20);
   }
```

```
  angleL = 60;
  angleR = 120;
  servo1.write(angleL);
  servo2.write(angleR);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
  {
    jaw.write(angleJ);
    delay(20);
  }
  for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
  {
    jaw.write(angleJ);
    delay(20);
  }
  angleL = 60;
  angleR = 120;
  servo1.write(angleL);
  servo2.write(angleR);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
  {
    jaw.write(angleJ);
    delay(20);
  }
  for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
  {
    jaw.write(angleJ);
    delay(20);
  }
  angleL = 60;
  angleR = 120;
  servo1.write(angleL);
  servo2.write(angleR);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 90; angleL++ && angleR-- &&
angleJ++)
  {
    jaw.write(angleJ);
    delay(20);
  }
  for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
  {
```

```
    jaw.write(angleJ);
    delay(20);
   }
   delay(1000);//1900 originally
   angleL = 90;
   angleR = 90;
   servo1.write(angleL);
   servo2.write(angleR);
   delay(200); //was originally 500

// dont tell gold i said that begins now
   angleL = 110;
   angleR = 70;
   angleJ = 70; //originally 30
   jaw.write(angleJ);
   servo1.write(angleL);
   servo2.write(angleR);
   sendCommand(CMD_PLAY_W_VOL, 0X1E01);

  for(angleJ=30 ; angleL < 80 && angleR > 100 && angleJ < 70; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 80 ; angleL > 80 && angleR < 100 && angleJ > 70; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
    delay(20);
   }
   angleL = 110;
   angleR = 70;
   servo1.write(angleL);
   servo2.write(angleR);
    for(angleJ=30 ; angleL < 80 && angleR > 100 && angleJ < 70; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 80 ; angleL > 80 && angleR < 100 && angleJ > 70; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
```

```
    delay(20);
   }
   angleL = 110;
   angleR = 70;
   servo1.write(angleL);
   servo2.write(angleR);
    for(angleJ=30 ; angleL < 80 && angleR > 100 && angleJ < 80; angleL++ && angleR-- &&
angleJ++)
   {
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 80 ; angleL > 80 && angleR < 100 && angleJ > 70; angleL-- && angleR++ &&
angleJ--)
   {
    jaw.write(angleJ);
    delay(20);
   }
   delay(500);// orginally 900
   angleL = 90;
   angleR = 90;
   angleJ = 80;
   jaw.write(angleJ);
   servo1.write(angleL);
   servo2.write(angleR);
   delay(1000);
 }

 // Thermistor Trigger
  Vo = analogRead(ThermistorPin);
  R2 = R1 * (1023.0 / (float)Vo - 1.0);
  logR2 = log(R2);
  T = (1.0 / (c1 + c2*logR2 + c3*logR2*logR2*logR2));
  T = T - 273.15;
  T = (T * 9.0)/ 5.0 + 32.0;
 if (T >= 78){
   angleL = 60;
   angleR = 120;
   servo1.write(angleL);
   servo2.write(angleR);
   sendCommand(CMD_PLAY_W_VOL, 0X1E03);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 80; angleL++ && angleR-- &&
angleJ++)
   {
    servo1.write(angleL);
```

```
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
  }
  for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
  {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
  }
  angleL = 60;
  angleR = 120;
  servo1.write(angleL);
  servo2.write(angleR);
  for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 80; angleL++ && angleR-- &&
angleJ++)
  {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
  }
  for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
  {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
  }

  delay(1000);

  angleL = 60;
  angleR = 120;
  servo1.write(angleL);
  servo2.write(angleR);
  for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 80; angleL++ && angleR-- &&
angleJ++)
  {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
```

```
    delay(20);
   }
   for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
   {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
   }
   angleL = 60;
   angleR = 120;
   servo1.write(angleL);
   servo2.write(angleR);
   for(angleJ=60 ; angleL < 80 && angleR > 100 && angleJ < 80; angleL++ && angleR-- &&
angleJ++)
   {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
   }
   for( angleJ = 90 ; angleL > 80 && angleR < 100 && angleJ > 80; angleL-- && angleR++ &&
angleJ--)
   {
    servo1.write(angleL);
    servo2.write(angleR);
    jaw.write(angleJ);
    delay(20);
   }
   delay(5000);
  }
}

void sendCommand(int8_t command, int16_t dat)
{
  delay(20);
  Send_buf[0] = 0x7e; //starting byte
  Send_buf[1] = 0xff; //version
  Send_buf[2] = 0x06; //the number of bytes of the command without starting byte and ending
byte
  Send_buf[3] = command; //
  Send_buf[4] = 0x00;//0x00 = no feedback, 0x01 = feedback
  Send_buf[5] = (int8_t)(dat >> 8);//datah
  Send_buf[6] = (int8_t)(dat); //datal
```

```
  Send_buf[7] = 0xef; //ending byte
  for(uint8_t i=0; i<8; i++)//
  {
    mySerial.write(Send_buf[i]) ;
  }
}
```