



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Control Systems

(ECE2010)

Project Report

Title: *GESTURE CONTROLLED COMPUTER*

Team Members:

Mayank Ranjan *17BEC0314*

Ajay Singh Rathore *17BEC0371*

Prathamesh Phutane *17BEC0871*

Naman Chawla *17BEC0906*

A Project Report

*Submitted in fulfilment of the requirement for the
evaluation of J component for the subject ECE 2010
Control Systems*

By

*Naman Chawla, Mayank Ranjan, Prathamesh Phutane
and Ajay Singh Rathore*

SUBMITTED TO: PROF. Rajesh R

EXECUTIVE SUMMARY

Recently **Gesture controlled Laptops or computers** are getting very famous. This technique is called Leap motion which enables us to control certain functions on our computer/Laptop by simply waving our hand in front of it. It is very cool and fun to do it, but these laptops are really priced very high. So, in this project let us try building our own **Gesture control Laptop/Computer by combining the Power of Arduino and Python.**

We will use two **Ultrasonic sensors** to determine the position of our hand and control any application or software that we like.

ACKNOWLEDGEMENT

In performing our assignment, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. We would like to show our gratitude to Prof. Rajesh R for giving us a good guideline for assignment throughout numerous consultations.

We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our assignment.

MOTIVATION

We have chosen this project with an interest of learning the direct interaction of humans with the consumer electronic devices. This takes the user experience to a whole new level. The gesture control technology would reduce our dependence on the age-old peripheral devices hence it would reduce the overall complexity of the system. Initially this technology was considered in the field of gaming (like Xbox Kinect), but the application of motion/gesture control technology would be more diverse if we apply it to our other electronics like computers, televisions, etc., for our day to day purposes like scrolling, selecting, clicking etc.

Our primary objective in doing this project was to build a device inspired from Leap motion. It is a device which recognizes hand gestures and can be used to virtually control a computer. In short, it provides a virtual screen with which we can interact with the computer. But the required hardware for making a device on these lines was not feasible, in terms of budget and time frame provided. So, we decided to build an introductory software implementation of the device which would eventually act as a virtual mouse.

INTRODUCTION

The concept behind the project is very simple. We will place two Ultrasonic (US) sensors on top of our monitor and will read the distance (after certain calibrations) between the monitor and our hand using Arduino, based on this value of distance we will perform certain actions. To perform actions on our computer we use Python “PyAutoGui” library.

The commands from Arduino are sent to the computer through serial port (USB) for this COM7 was used. This data will be then read by python which is running on the computer and based on the read data an action will be performed.

To control the PC with Hand Gestures, just connect the two Ultrasonic sensors with Arduino. We know US sensor work with 5V and hence they are powered by the on-board Voltage regulator of Arduino. The Arduino can be connected to the PC/Laptop for powering the module and also for Serial communication.

Once the connections are done place them on your monitor. We have used a double side tape to stick it to the monitor and newspaper to cover the back so as to protect it from scratches. After securing it in a place we can proceed with the Programming.

Programming the Arduino:

The Arduino should be programmed to read the distance of hand from the US sensor. The **complete program** is given after this page.

By reading the value of distance we can arrive at certain actions to be controlled with gestures, for example in this program We have programmed **7 actions** as a demo.

Action 1: When both the hands are placed up before the sensor at a particular far distance then the video in VLC player should Play/Pause.

Action 2: When right hand is placed up before the sensor at a particular far distance then the video should Fast Forward one step.

Action 3: When left hand is placed up before the sensor at a particular far distance then the video should Rewind one step.

Action 4: When right hand is placed up before the sensor at a particular near distance and then if moved towards the sensor the video should fast forward and if moved away the video should Rewind.

Action 5: When left hand is placed up before the sensor at a particular near distance and then if moved towards the sensor the volume of video should increase and if moved away the volume should Decrease.

Action 6: When both the hands are in very close range of ultrasonic sensors (I.e.less than 5cms.) then the application should get closed.

Action 7: When left hand is placed within range of 30cms, moving the hand to and fro yields in scrolling any document up and down respectively.

Arduino Code for video controlling and scrolling:



```
const int trigger1 = 3; //Trigger pin of 1st Sesnor
const int echo1 = 5; //Echo pin of 1st Sesnor
const int trigger2 = 6; //Trigger pin of 2nd Sesnor
const int echo2 = 9; //Echo pin of 2nd Sesnor

long time_taken;
int dist, distL, distR;

void setup() {
  Serial.begin(9600);

  pinMode(trigger1, OUTPUT);
  pinMode(echo1, INPUT);
  pinMode(trigger2, OUTPUT);
  pinMode(echo2, INPUT);
}

/*###Function to calculate distance###*/
void calculate_distance(int trigger, int echo)
{
  digitalWrite(trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);

  time_taken = pulseIn(echo, HIGH);
  dist= time_taken*0.034/2;
  if (dist>50)
    dist = 50;
```


Pause / Play:

```
}

void loop() { //infinite loopy
  calculate_distance(trigger1,echo1);
  distL =dist; //get distance of left sensor

  calculate_distance(trigger2,echo2);
  distR =dist; //get distance of right sensor

  //Uncomment for debudding
  /*Serial.print("L=");
  Serial.println(distL);
  Serial.print("R=");
  Serial.println(distR);
  */

  //Pause Modes -Hold
  if ((distL >30 && distR>30) && (distL <50 && distR<50)) //Detect both hands
  {Serial.println("Play/Pause"); delay (500);}

  calculate_distance(trigger1,echo1);
  distL =dist;

  calculate_distance(trigger2,echo2);
  distR =dist;
  if((distL>40 && distR>40) && (distL<50 && distR<50)){
    Serial.println("Play/Pause"); delay (500);
  }
}
```

Volume up/down, Page up/down:

```
//Control Modes
//Lock Left - Control Mode
if (distL>=13 && distL<=17)
{
    delay(100); //Hand Hold Time
    calculate_distance(trigger1,echo1);
    distL =dist;
    if (distL>=13 && distL<=17)
    {
        Serial.println("Left Locked");
        while(distL<=30)
        {
            calculate_distance(trigger1,echo1);
            distL =dist;
            if (distL<10) //Hand pushed in
            {Serial.println ("Vup"); delay (300);}
            {Serial.println ("Pup"); delay (300);}
            if (distL>20) //Hand pulled out
            {Serial.println ("Vdown"); delay (300);}
            {Serial.println ("Pdown"); delay (300);}
        }
    }
}
```

Closing any application/software:

```
//Lock both - Control Mode
if ((distL <=5 && distR<=5)//Detect both hands
{Serial.println("CLOSE"); delay (500);}

calculate_distance(trigger1,echo1);
distL =dist;

calculate_distance(trigger2,echo2);
distR =dist;
if((distL <=5 && distR<=5){
    Serial.println("CLOSE"); delay (500);
}
```

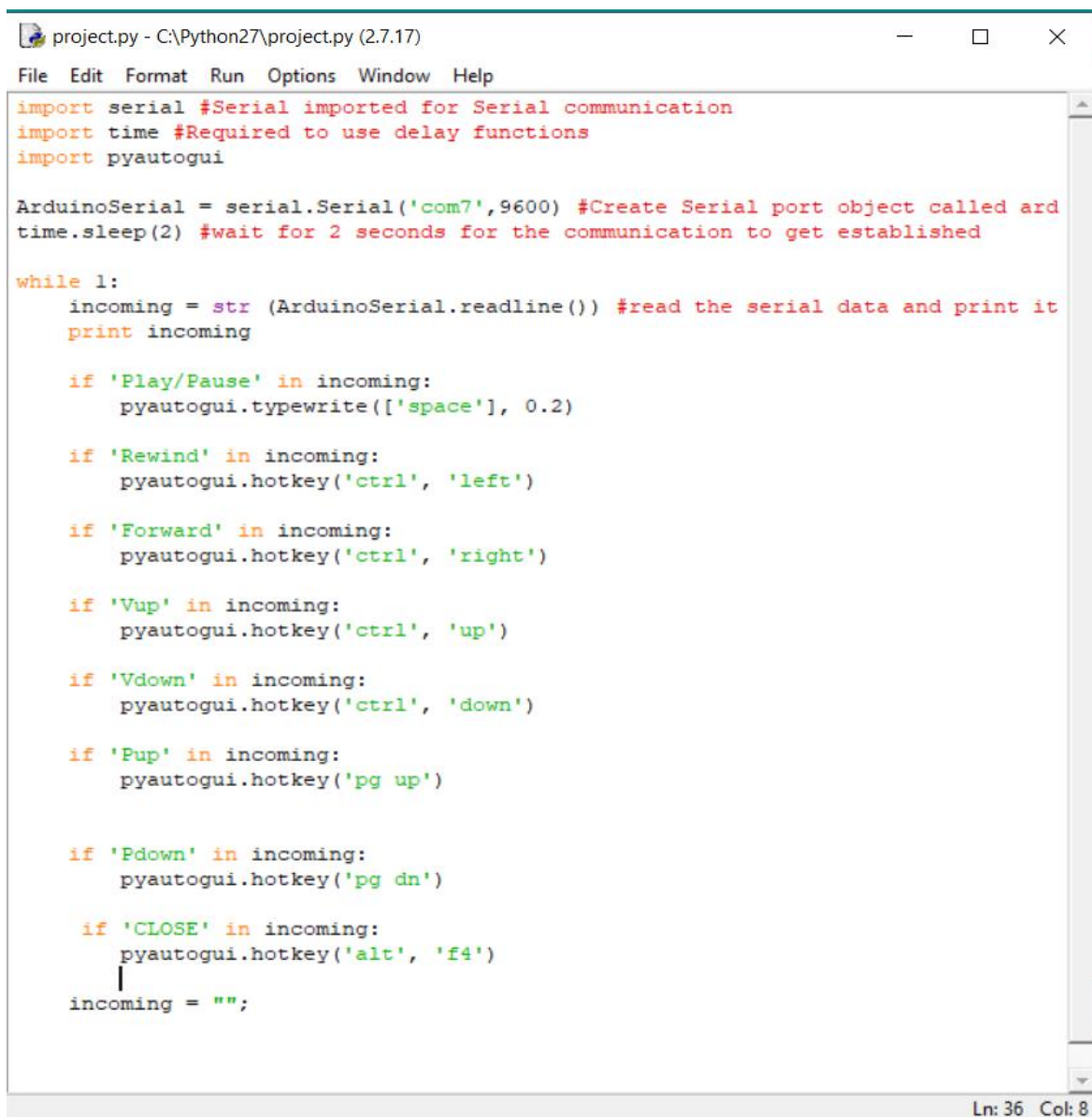
Rewind/forward:

```
//Lock Right - Control Mode
if (distR>=13 && distR<=17)
{
    delay(100); //Hand Hold Time
    calculate_distance(trigger2,echo2);
    distR =dist;
    if (distR>=13 && distR<=17)
    {
        Serial.println("Right Locked");
        while(distR<=40)
        {
            calculate_distance(trigger2,echo2);
            distR =dist;
            if (distR<10) //Right hand pushed in
            {Serial.println ("Rewind"); delay (300);}
            if (distR>20) //Right hand pulled out
            {Serial.println ("Forward"); delay (300);}
        }
    }
}

delay(200);
}
```

Programming in Python:

The python program for this project is very simple. We just have to establish a serial communication with Arduino through the correct baud rate (in this case the baud rate is 9600) and then perform some basic keyboard actions. The first step with python is to install the **pyautogui** module. It is very vital otherwise the **program will not work without pyautogui module**. This module helps the user to control the keys of keyboard and mouse. This makes it so important in our project. It reads data from serial communication and the corresponding action takes place. The following code shows which command uses which keys:



```
project.py - C:\Python27\project.py (2.7.17)
File Edit Format Run Options Window Help

import serial #Serial imported for Serial communication
import time #Required to use delay functions
import pyautogui

ArduinoSerial = serial.Serial('com7',9600) #Create Serial port object called ard
time.sleep(2) #wait for 2 seconds for the communication to get established

while 1:
    incoming = str (ArduinoSerial.readline()) #read the serial data and print it
    print incoming

    if 'Play/Pause' in incoming:
        pyautogui.typewrite(['space'], 0.2)

    if 'Rewind' in incoming:
        pyautogui.hotkey('ctrl', 'left')

    if 'Forward' in incoming:
        pyautogui.hotkey('ctrl', 'right')

    if 'Vup' in incoming:
        pyautogui.hotkey('ctrl', 'up')

    if 'Vdown' in incoming:
        pyautogui.hotkey('ctrl', 'down')

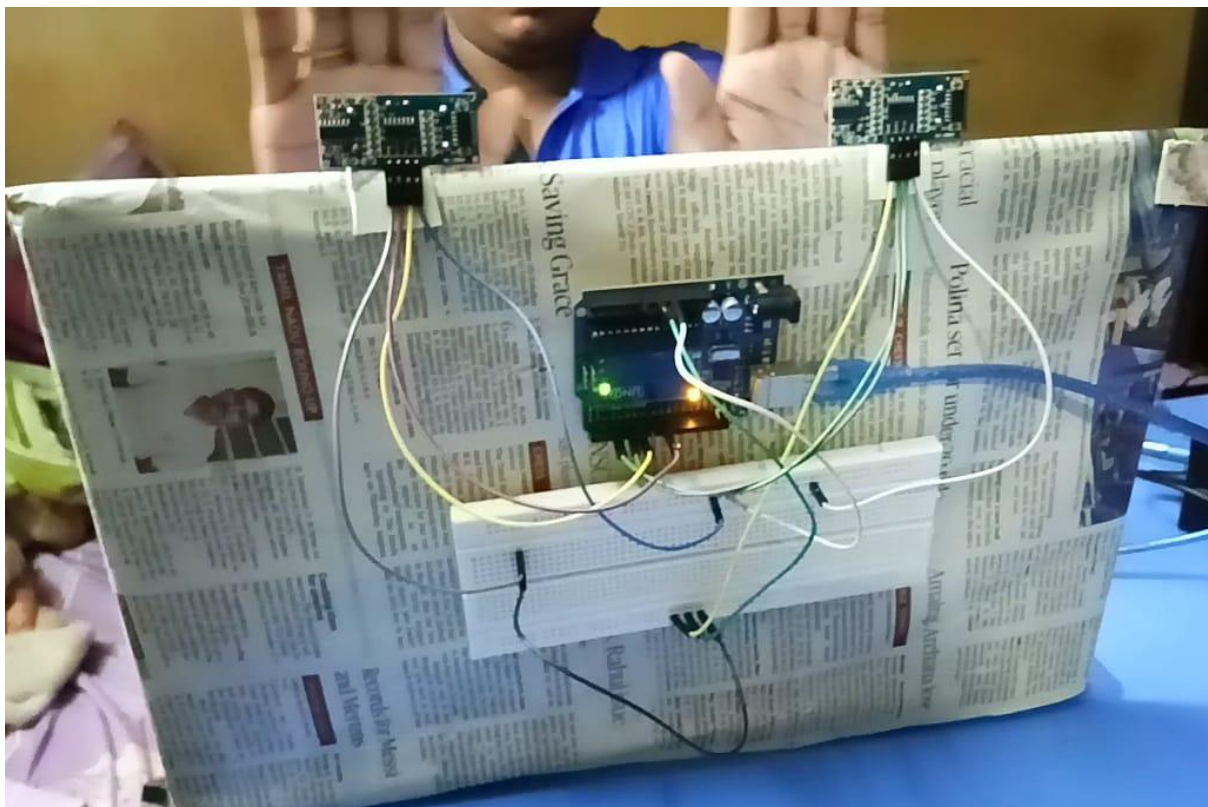
    if 'Pup' in incoming:
        pyautogui.hotkey('pg up')

    if 'Pdown' in incoming:
        pyautogui.hotkey('pg dn')

    if 'CLOSE' in incoming:
        pyautogui.hotkey('alt', 'f4')
    incoming = ""
```

Ln: 36 Col: 8

Our Gesture Controlled Computer in Action:



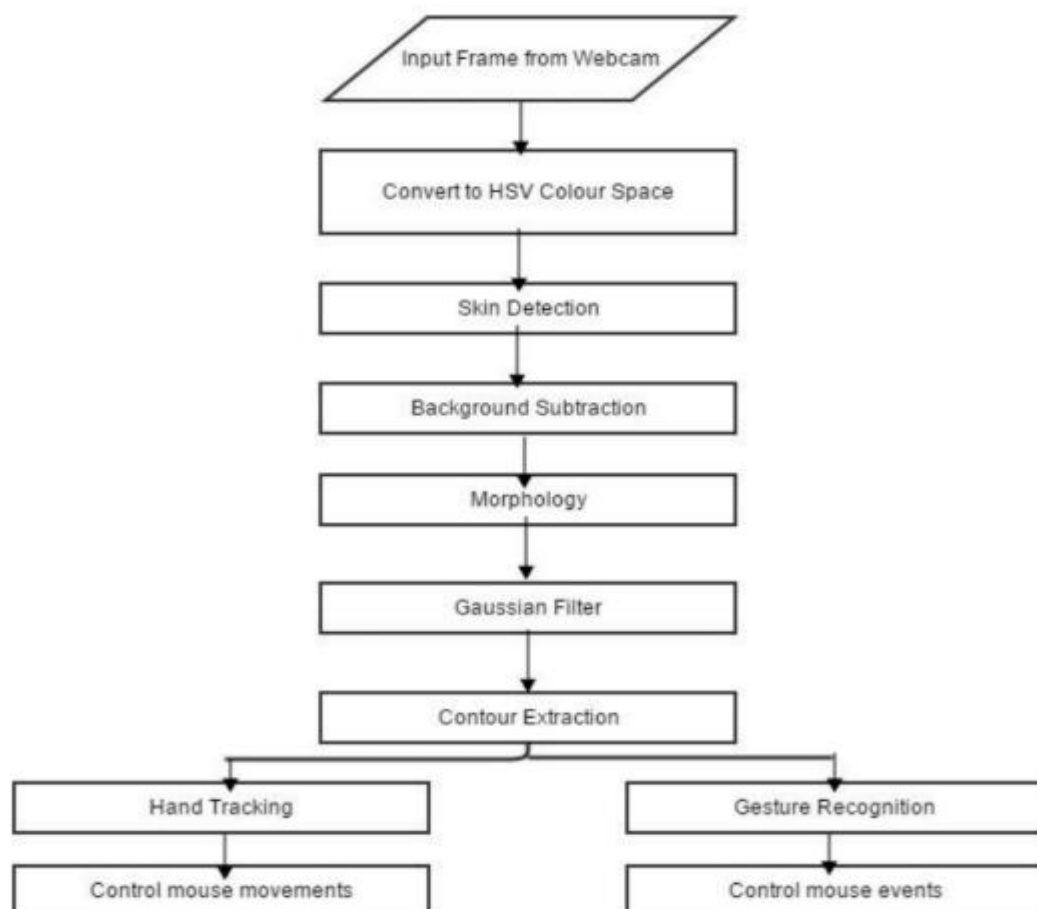
Working Prototype link:

<https://www.youtube.com/watch?v=n4iAfJpPQkc&feature=youtu.be>

Programming the Arduino for other types of gesture recognition:

The following section is the idea of **further improving** our prototype and **modifying** it such that it may not only be limited to distances but also recognize other gestures like swiping the palm will do slide-show, closing a fist in front of web-cam will zoom out and the reverse action will do zoom in etc.

In this section the strategies and methods used in the design and development of the vision-based systems are explained. The algorithm for the entire system is shown in Figure below.



ISSUES TO HAND GESTURE RECOGNITION: EXTRACTION METHODS AND FEATURES EXTRACTION

Segmentation process is the first process for recognizing hand gestures. It is the process of dividing the input image (in this case hand gesture image) into regions separated by boundaries. The segmentation process depends on the type of gesture, if it is dynamic gesture then the hand gesture need to be located and tracked, if it is static gesture (posture) the input image have to be segmented only. The hand should be located firstly, generally a bounding box is used to specify the depending on the skin color and secondly, the hand have to be tracked, for tracking the hand there are two main approaches; either the video is divided into frames and each frame have to be processed alone, in this case the hand frame is treated as a posture and segmented or using some tracking information such as shape, skin color using some tools such as Kalman filter.

The common helpful cue used for segmenting the hand is the skin color, since it is easy and invariant to scale, translation, and rotation changes. Different tools and methods used skin and non-skin pixels to model the hand. These methods are parametric and non-parametric techniques, Gaussian Model (GM) and Gaussian Mixture Model (GMM) are parametric techniques, and histogram based techniques are non-parametric. However it is affected with illumination condition changes abs different races. Some researches overcome this problem using data glove and colored markers which provide exact information about the orientation and position of palm and fingers. Others used infrared camera, and range information generated by special camera Time-of-Flight (ToF) camera, although these systems can detect different skin colors under cluttered background but it is affected with changing in temperature degrees besides their expensive cost.

Gestures Classification

After modeling and analysis of the input hand image, gesture classification method is used to recognize the gesture. Recognition process affected with the proper selection of features parameters and suitable classification algorithm. For example edge detection or contour operators cannot be used for gesture recognition since many hand postures are generated and produce miss-classification . Euclidean distance metric used to classify the gestures Statistical tools used for gesture classification, HMM tool has shown its ability to recognize dynamic gestures besides, Finite State Machine (FSM), Learning Vector Quantization , and Principal Component Analysis (PCA). Neural network has been widely applied in the field of extracted the hand shape, and for hand gesture recognition Other soft computing tools are effective in this field as well, such as Fuzzy C Means clustering (FCM), and Genetic Algorithms GAs .

APPLICATION AREAS OF HAND GESTURES SYSTEM

Hand gestures recognition system has been applied for different applications on different domains, as mentioned in including; sign language translation, virtual environments, smart surveillance, robot control, medical systems etc. overview of some hand gesture application areas are listed below

A. Sign Language Recognition:

Since the sign language is used for interpreting and explanations of a certain subject during the conversation, it has received special attention. A lot of systems have been proposed to recognize gestures using different types of sign languages . For example recognized American Sign Language ASL using boundary histogram, MLP neural network and dynamic programming matching recognized Japanese sign language JSL using Recurrent Neural Network, 42 alphabet and 10 words. recognized Arabic Sign language ArSL using two different types of Neural Network, Partially and Fully Recurrent neural Network.

B. Robot Control:

Controlling the robot using gestures considered as one of the interesting applications in this field proposed a system that uses the numbering to count the five fingers for controlling a robot using hand pose signs. The orders are given to the robot to perform a particular task, where each sign has a specific meaning and represents different function for example, “one” means “move forward”, “five” means “stop”, and so on

C. Graphic Editor Control:

Graphic editor control system requires the hand gesture to be tracked and located as a pre-processing operation. 12 dynamic gestures can be used for drawing and editing graphic system. Shapes for drawing are; triangle, rectangular, circle, arc, horizontal and vertical line for drawing, and commands for editing graphic system are; copy, delete, move, swap, undo, and close.

D. Virtual Environments (VEs):

One of the popular applications in gesture recognition system is virtual environments VEs, especially for communication media systems provided 3D pointing gesture recognition for natural human computer Interaction HCI in a real-time from binocular views. The proposed system is accurate and independent of user characteristics and environmental changes

E. Numbers Recognition:

Another recent application of hand gesture is recognizing numbers. An automatic system has also been proposed that could isolate and recognize a meaningful gesture from hand motion of Arabic numbers from 0 to 9 in a real time system using HMM.

F. Television Control:

Hand postures and gestures are used for controlling the Television device in a set of hand gesture are used to control the TV activities, such as turning the TV on and off, increasing and decreasing the volume, muting the sound, and changing the channel using open and close hand .

Conclusion

The concept behind the project is very simple. Just placing two Ultrasonic (US) sensors on top of our monitor and reading the distance between the monitor and our hand using Arduino can alone do a major part of controlling any computer. All of this is done using basic calibrations and distance measurement. To perform actions on our computer we did Python coding, Arduino coding and PyAutoGui library was used. The commands from Arduino are sent to the computer through serial port (USB). This data will be then read by python which is running on the computer and based on the read data an action will be performed.