DIY IIT Kharagpur

Gesture Controlled Media Player

Team 7

Team Composition

YASH RANA (20PH20045)

SNEHA KUMARI (20IM10030)

DEEPA RAJ (20IM10006)

G CHETHAN (20ME30021)

CONTENTS

1.	**Acknowledgement**
2.	**Introduction**
3.	**Working principle**
4.	**Components**
5.	**Actions enabled**
6.	**Circuit: Simulated v/s Implemented**
7.	**Code execution: Arduino**
8.	**Flowchart: Arduino**
9.	**Code execution: Python**
10.	**Future scope of our project**
11.	**Work distribution**
12.	**Challenges faced**
13.	**Bibliography**

ACKNOWLEDGEMENT

Our 1st project of arduino “controlling media player using hand gestures” had a lot for us to learn . We would really like to thank our DIY course coordinators for giving us such a great opportunity. We would also like to thank our mentor prof. and our TA who helped and cleared our doubts for this project and hence we completed this project with great success.

Thank you to the professors and the team.

Introduction

Gesture control can be defined as the ability to recognize and interpret movements of the human body in order to interact with and control a computer system without direct physical contact. The term “natural user interface” is becoming commonly used to describe these interface systems, reflecting the general lack of any intermediate devices between the user and the system.

The concept behind the project is pretty simple. Two Ultrasonic (US) sensors were placed on top of our monitor and read the distance and it’s change between the monitor and our hand, recorded through Arduino. Based on this value of distance certain actions were performed through Python code.

The **PyAutoGUI** library of Python was utilized to perform this task. It let our Python scripts control the mouse and keyboard to automate interactions with other applications. Documentation: https://pyautogui.readthedocs.io/en/latest/

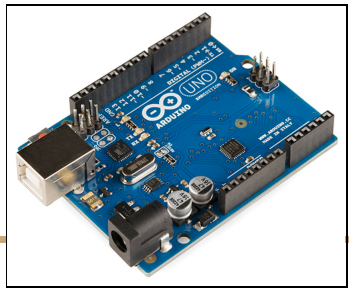
Working principle:

The commands from Arduino are sent to the computer through serial port (USB). This data is then read by python which is running on the computer and based on the read data specific actions will be performed.

Components required:

The components used in making this project are as follows:

1. **Arduino UNO** : The Arduino Uno is an open-source microcontroller board based on the Microchip



ATmega328P microcontroller and developed by Arduino.cc

2. **Ultrasonic Distance Sensors** : An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal.Used to measure the distance of an object placed in front of the sensor.

3. **Jumper wires**: Connecting wires to complete the arduino circuit.

4. **Laptop/PC**: A laptop with decent specifications to run python.

Actions enabled

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

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

Action 4: When the 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 the 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.

Circuit: Simulated v/s Real

Following are two images, one representing the simulated circuit, and the other representing the actual implemented circuit.

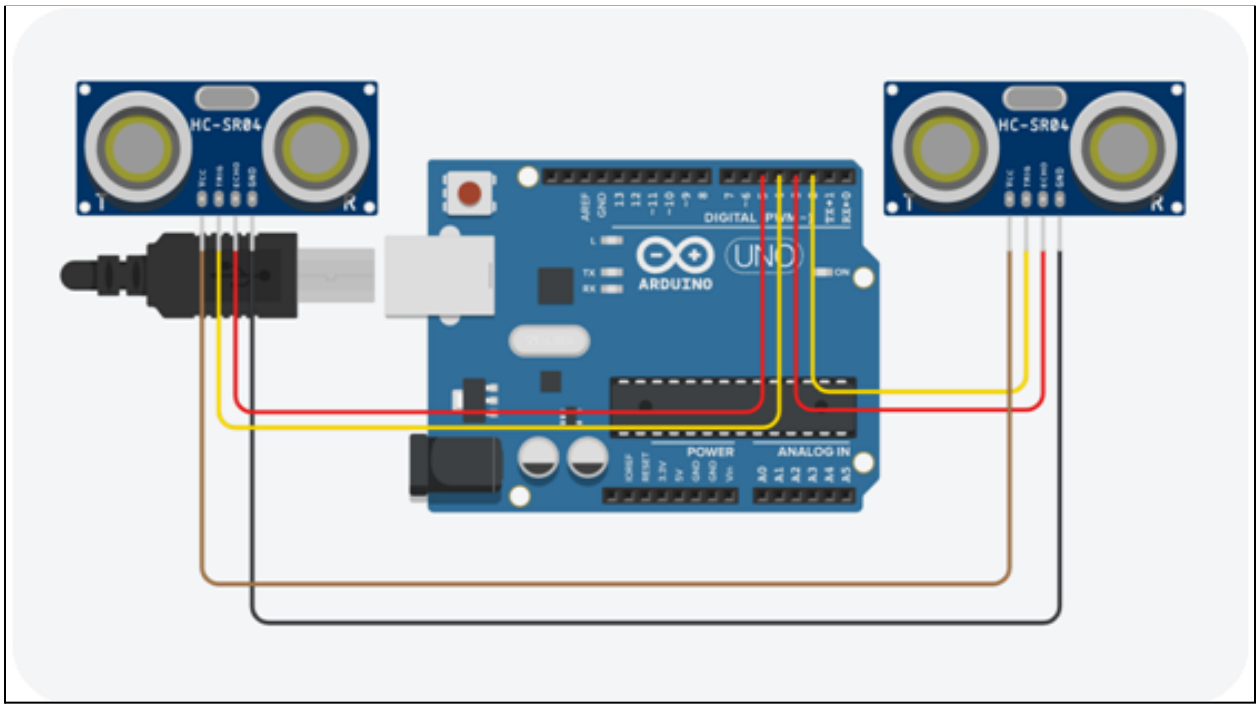


Fig. TinkerCAD Simulated Circuit.

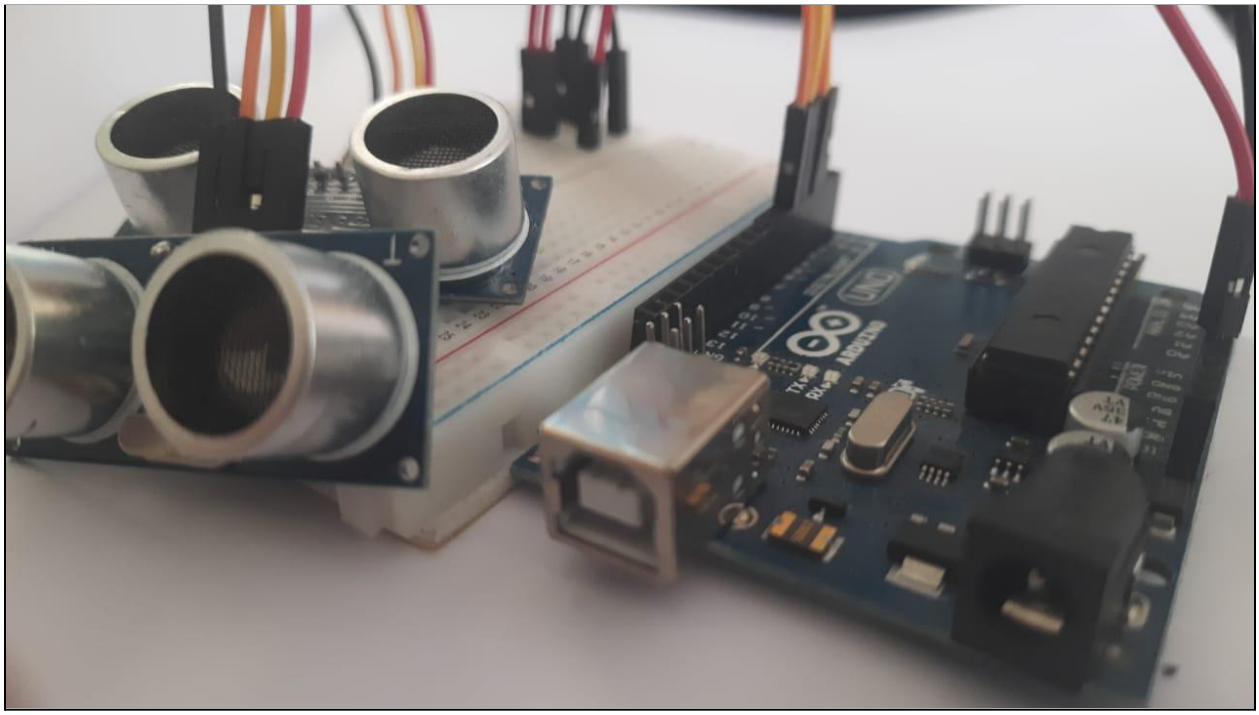


Fig. Real circuit

Code execution:

Let us see how the program is written to perform the required actions.

We start with defining the I/O pins.

The two US sensors are connected to Digital pins 2,3,4 and 5 and are powered by +5V pins. The Trigger pins are output pins and Echo pins are input pins.

The code will have two sections. One which runs the Arduino and the other part which communicates the Arduino outputs to VLC using Python 3.7.

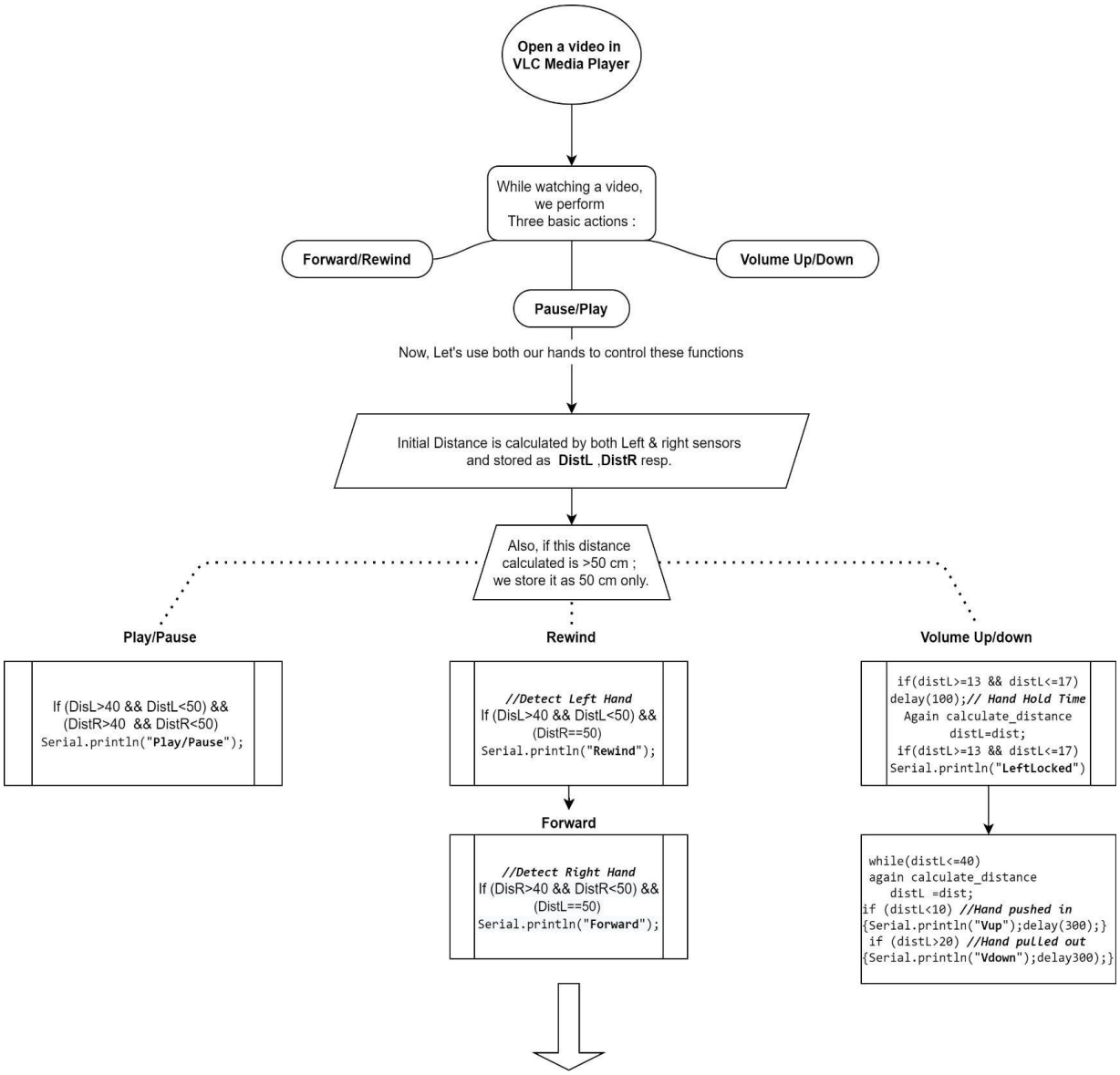
Part 1: Arduino

The Serial communication between Arduino and python takes places at a baud rate of 9600.

```
const int trigger1 = 2; //Trigger pin of 1st Sensor
const int echo1 = 3; //Echo pin of 1st Sensor
const int trigger2 = 4; //Trigger pin of 2nd Sensor
const int echo2 = 5;//Echo pin of 2nd Sensor
void setup() {
Serial.begin(9600);
pinMode(trigger1, OUTPUT);
pinMode(echo1, INPUT);
pinMode(trigger2, OUTPUT);
pinMode(echo2, INPUT);
}


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;
}

calculate_distance(trigger1,echo1);
distL =dist; //get distance of left sensor
calculate_distance(trigger2,echo2);
distR =dist; //get distance of right sensor

if ((distL >40 && distR>40) && (distL <50 && distR<50))
{Serial.println("Play/Pause"); delay (500);}
if ((distL >40 && distL<50) && (distR ==50))
{Serial.println("Rewind"); delay (500);}

if ((distR >40 && distR<50) && (distL ==50))
{Serial.println("Forward"); delay (500);}

//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<=40)
    {
      calculate_distance(trigger1,echo1);
      distL =dist;
      if (distL<10) //Hand pushed in
      {Serial.println ("Vup"); delay (300);}
      if (distL>20) //Hand pulled out
      {Serial.println ("Vdown"); delay (300);}
    }
  }
}
```



Now we repeatedly listen to the port and compare them to predefined words and perform functions accordingly.

```
print incoming

if 'Play/Pause' in incoming:
        pyautogui.typewrite(['space'],0.2) #press spacebar for 0.2 secs
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')
```

Fig. Flowchart of the Arduino code

Part 2: Python

We had to establish a serial communication with Arduino through the correct baud rate and then perform some basic keyboard actions. The first step with python would be to install the pyautogui module.

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

ArduinoSerial = serial.Serial('com18',9600) #Create Serial port object called arduinoSerialData
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 as line
    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', 'down')

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

incoming = "";
```

Future scope of our project and how this tech could benefit us

Our project which involves using gestures to control laptops could be applied to anything which would ease our lifestyle , it could be something as big as controlling a car to something as small as switching off the lights at home . In our project we have used ultrasonic sound sensors to detect motion in gestures but this could be taken to another step by using computer vision . Now why computer vision steps up in the game is because it can take input from 2 dimensional images whereas the ultrasonic sensors would just take one dimensional distance as input . Gesture recognition technology if perfected can be used to render input devices such as keyboard , mouse and touch screens redundant.While it might seem like a technology that will only increase our lethargy , truth is that other than making life easier and look cool , it also has a lot of applications in almost all fields.

How this benefits us :

Medical Applications – Advanced robotics systems with gesture recognition can be placed in hospitals or homes to recognize and treat life threatening conditions like heart attacks or strokes.

Alternative computer interfaces – Gesture recognition, along with voice recognition, facial recognition, lip movement recognition and eye tracking combined can be used to create something called a perceptual user interface (PUI), a completely different way to

interact with computer systems which will improve usability and creativity by leaps and bounds.

Entertainment applications – Most video games today are played either on game consoles, arcade units or PCs, and all require a combination of input devices. Gesture recognition can be used to truly immerse players in the game world like never before.

Automation systems – In homes, offices, transport vehicles and more, gesture recognition can be incorporated to greatly increase usability and reduce the resources necessary to create primary or secondary input systems like remote controls, car entertainment systems with buttons or similar.

An easier life for the disabled – One of the biggest challenges faced today is providing separate and equally non cumbersome services to the differently abled and handicapped. While there are special provisions around the world, there's still huge room for improvement to bring all lives on equal footing. Gesture recognition technology can eliminate a lot of manual labor and make life much easier for those who aren't as fortunate as most of us are.

WORK DISTRIBUTION

- Deepa worked on completing the tinker cad model.
- Deepa and Sneha worked on Arduino code.
- Chethan worked on the Python code along with Yash and also constructed the hardware model.
- Sneha and Yash worked on finishing the track control and forward/rewind function.
- Testing for this project was done by Chethan.
- Yash,Chethan,Sneha and Deepa are responsible for putting together the report for this project.

Challenges faced:

1. We initially had some difficulty in figuring out the circuit arrangement but Yash and Deepa thought of an easy and efficient solution to the problem.

2. We faced problems in the code setup while installing pyserial and other PIP tools. This was solved by Chethan and Yash by installing a newer version of the Py module.

3. We faced problems in single step forward/rewind which were solved by Sneha.

Here's the YouTube link of our project video:

https://youtu.be/eQWyaTrtFak

-Credits: G Chethan

BIBLIOGRAPHY

- Getting started: https://www.arduino.cc/en/Guide
- PyAutoGUI: https://pypi.org/project/PyAutoGUI/
- https://en.wikipedia.org/wiki/Gesture_recognition
- Advance Python programming: https://youtu.be/1RE5tSPO2RI