ARDUINO BASED HAND GESTURE CONTROL OF YOUR COMPUTER

by

SIDDHARTH S 17BEC1042

KISHORE NITHIN S 17BEC1068

RAMNATH S 17BEC1107

SOWDESHWAR S 17BEC1196

A project report submitted to

**Prof. RAMESH. R**

**SCHOOL OF ELECTRONICS ENGINEERING**

in partial fulfilment of the requirements for the course of

**ECE1005 – SENSORS AND INSTRUMENTATION**

in

**B.TECH. ELECTRONICS AND COMMUNICATION ENGINEERING**



**VIT UNIVERSITY, CHENNAI**

**Vandalur – Kelambakkam Road**

**Chennai – 600127**

**MARCH 2018**

**BONAFIDE CERTIFICATE**

Certified that this project report entitled “**ARDUINO BASED HAND GESTURE CONTROL OF YOUR COMPUTER**”is a Bonafide work of **RAMNATH S(17BEC1107), SIDDHARTH S(17BEC1042), KISHORE NITHIN S(17BEC1068)** and **SOWDESHWAR S(17BEC1196)**.

**Dr. VETRIVELAN.P**

Associate Professor

School of Electronics Engineering (SENSE),

VIT University, Chennai

Chennai – 600 127.

**ABSTRACT**

Gesture control is a very interesting phenomenon which serves as an interface between the user and device. In recent years, this phenomenon has been brought to light of the technological world through the marvellous advancements in the tech world. Gesture control is achieved by a sensor device that supports hand and finger motions as input, analogous to a mouse, but requires no hand contact or touching.

Motion controllers using accelerometers are used as controllers for video games, which was made more popular since 2006 by the Wii Remote controller for Nintendo's Wii console, which uses accelerometers to detect its approximate orientation and acceleration, and serves an image sensor, so it can be used as a pointing device. A very popular device in this regard is the Kinect by Microsoft. Kinect is a line of motion sensing input devices that was produced by Microsoft for Xbox 360 and Xbox One video game consoles and Microsoft Windows PCs. Based around a webcam-style add-on peripheral, it enables users to control and interact with their console/computer without the need for a game controller, through a natural user interface using gestures and spoken commands. The device features an "RGB camera, depth sensor and multi-array microphone running proprietary software", which provide full-body 3D motion capture, facial recognition and voice recognition capabilities.

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 user-friendly, but these laptops are really expensive.

In this project, we have implemented a simple Arduino based hand gesture control where you can control few functions of your web browser like switching between tabs, scrolling up and down in web pages, shift between tasks (applications), play or pause a video and increase or decrease the volume (in VLC Player) with the help of hand.

The position of the Ultrasonic Sensors is very important. Place the two Ultrasonic Sensors on the top of a laptop screen at either end. The distance information from Arduino is collected by a Python Program and a special library called PyAutoGUI will convert the data into keyboard click actions.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **SERIAL NO.** |  | **NAME** | **PAGE NO.** |
|  |  | ABSTRACT | 3 |
|  |  | ACKNOWLEDGEMENT | 4 |
|  |  |  |  |
| 1 |  | INTRODUCTION | 6 |
|  |  |  |  |
|  | 1.1 | OBJECTIVES AND GOALS | 6 |
|  |  |  | 6 |
|  | 1.3 | FEATURES | 6 |
|  |  |  |  |
| 2 |  | DESIGN | 6-10 |
|  | 2.1 | CIRCUIT DIAGRAM | 6 |
|  | 2.2 | HARDWARE ANALYSIS | 7-10 |
|  |  |  |  |
| 3 | SOFTWARE | ANALYSIS | 11-14 |
|  | 3.1 | ARDUINO CODE | 11-13 |
|  | 3.2 | PYTHON CODE | 14 |
| 4 | 4.1 | RESULT, CONCLUSION | 15 |
|  | 4.2 | FUTURE WORK | 15 |
| 5 |  | REFERENCES | 16 |

1. **INTRODUCTION**

**1.1 OBJECTIVES AND GOALS**

In this project, we have implemented a simple Arduino based hand gesture control where you can control few functions of your web browser like switching between tabs, scrolling up and down in web pages, shift between tasks (applications), play or pause a video and increase or decrease the volume (in VLC Player) with the help of hand gestures.

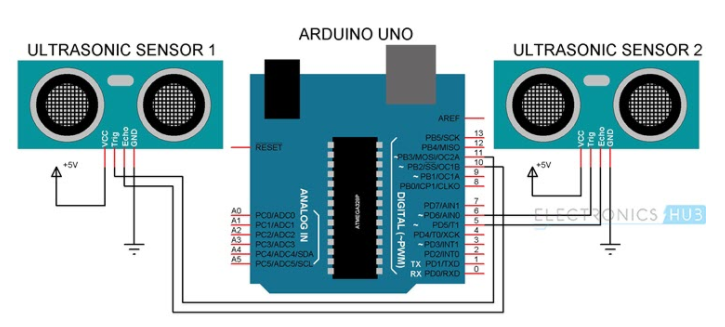
**1.2 FEATURES**

The hand gestures in front of the Ultrasonic sensors can be calibrated so that they can perform five different tasks on your computer.

* Switch to Next Tab in a Web Browser
* Scroll Down in a Web Page
* Scroll Up in a Web Page
* Switch between two Tasks (Chrome and VLC Player)
* Play/Pause Video in VLC Player
* Increase Volume
* Decrease Volume

**2 ARDUINO BASED HAND GESTURE CONTROL OF YOUR COMPUTER**

**2.1 CIRCUIT DIAGRAM**

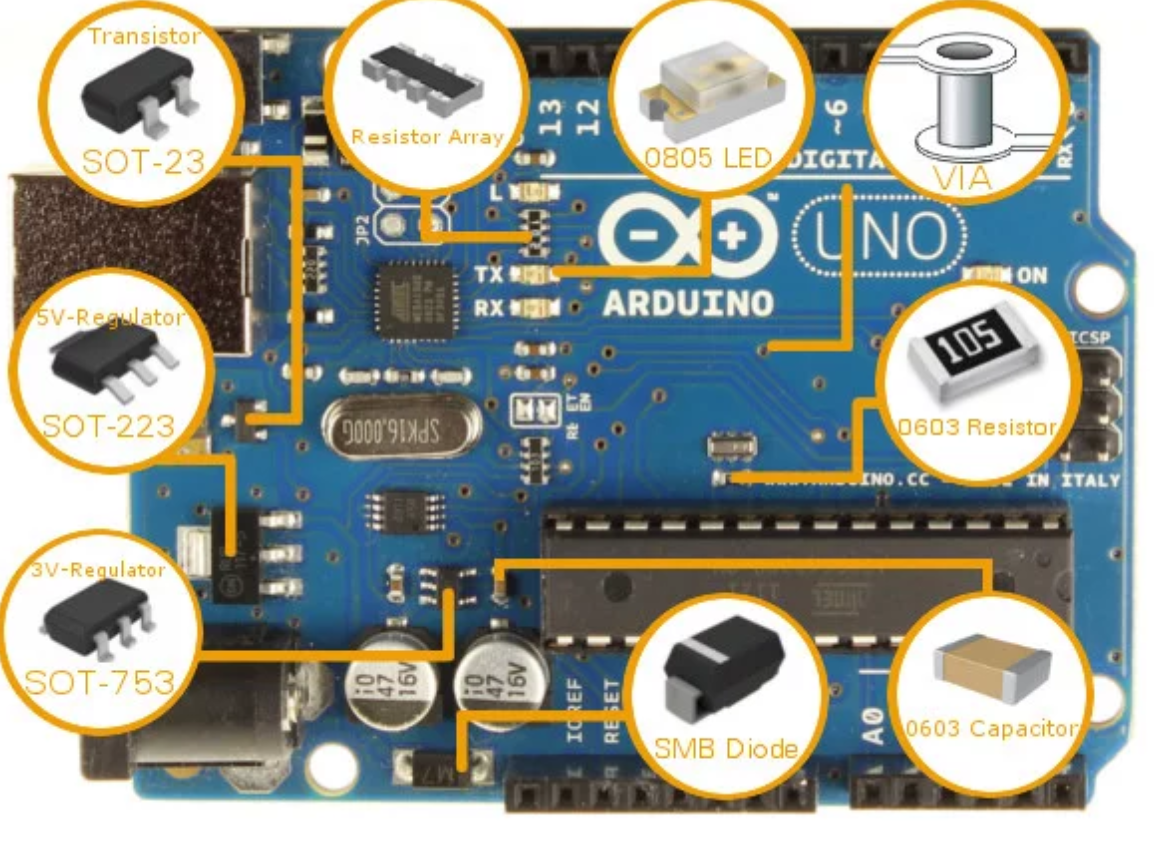


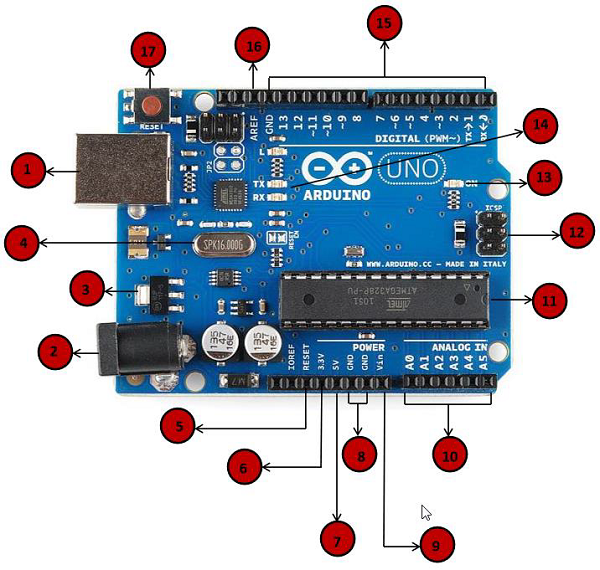
The two main features of the basic block diagram (given below) are

***HARDWARE ANALYSIS***

* ARDUINO UNO
* ULTRA SONIC SENSOR

**ARDUINO UNO**





|  |  |
| --- | --- |
|  | **Power USB**  Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1). |
|  | **Power (Barrel Jack)**  Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2). |
|  | **Voltage Regulator**  The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements. |
|  | **Crystal Oscillator**  The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz. |

|  |  |
| --- | --- |
|  | **Arduino Reset**  You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5). |
|  |  |
|  | **Analog pins**  The Arduino UNO board has five analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor. |
|  | **Main microcontroller**  Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet. |
|  | **ICSP pin**  Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus. |
|  | **Power LED indicator**  This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection. |
|  | **TX and RX LEDs**  On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process. |
|  | **Digital I/O**  The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM. |
|  | **AREF**  AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins. |

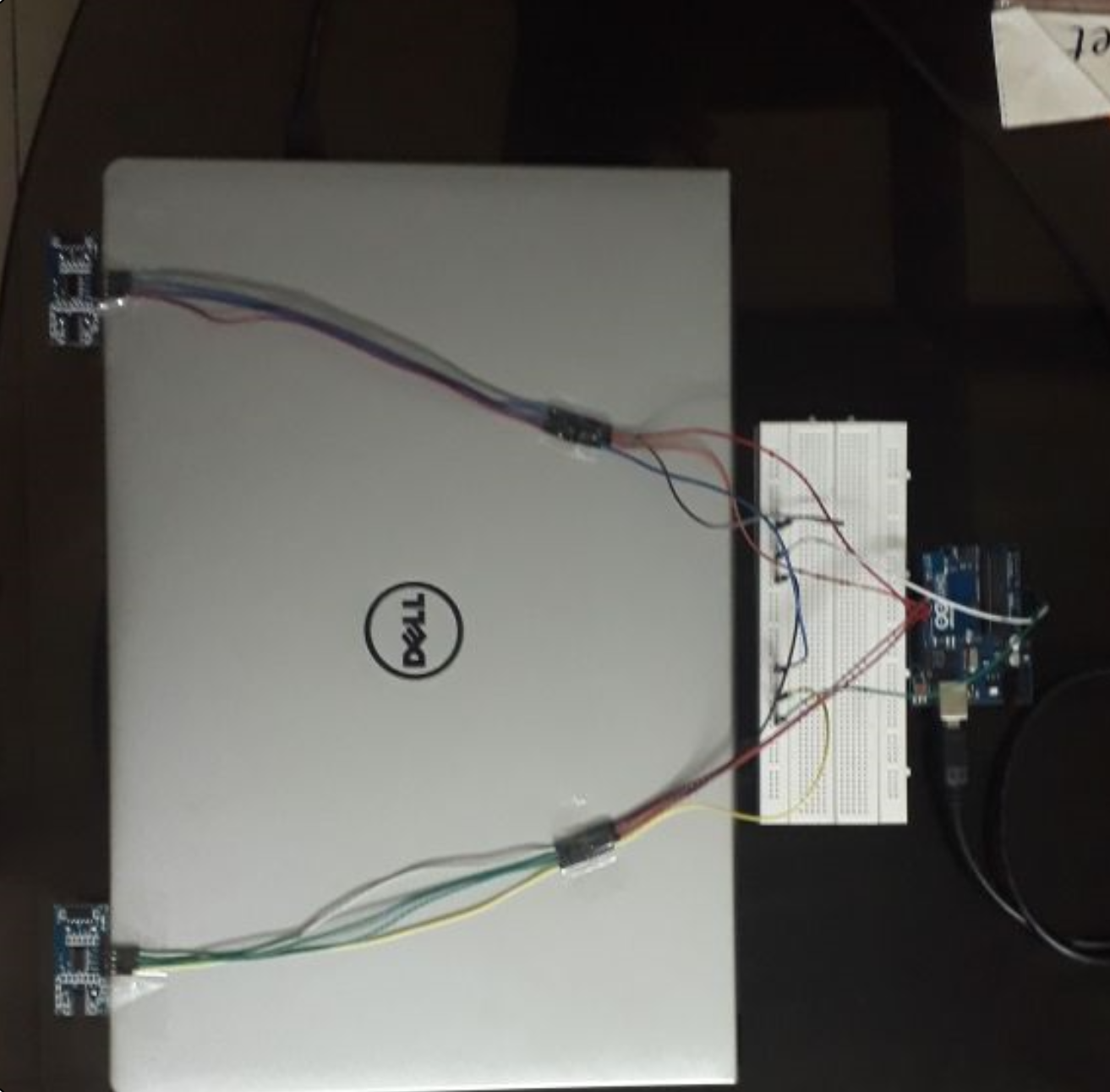
**ULTRASONIC SENSOR**

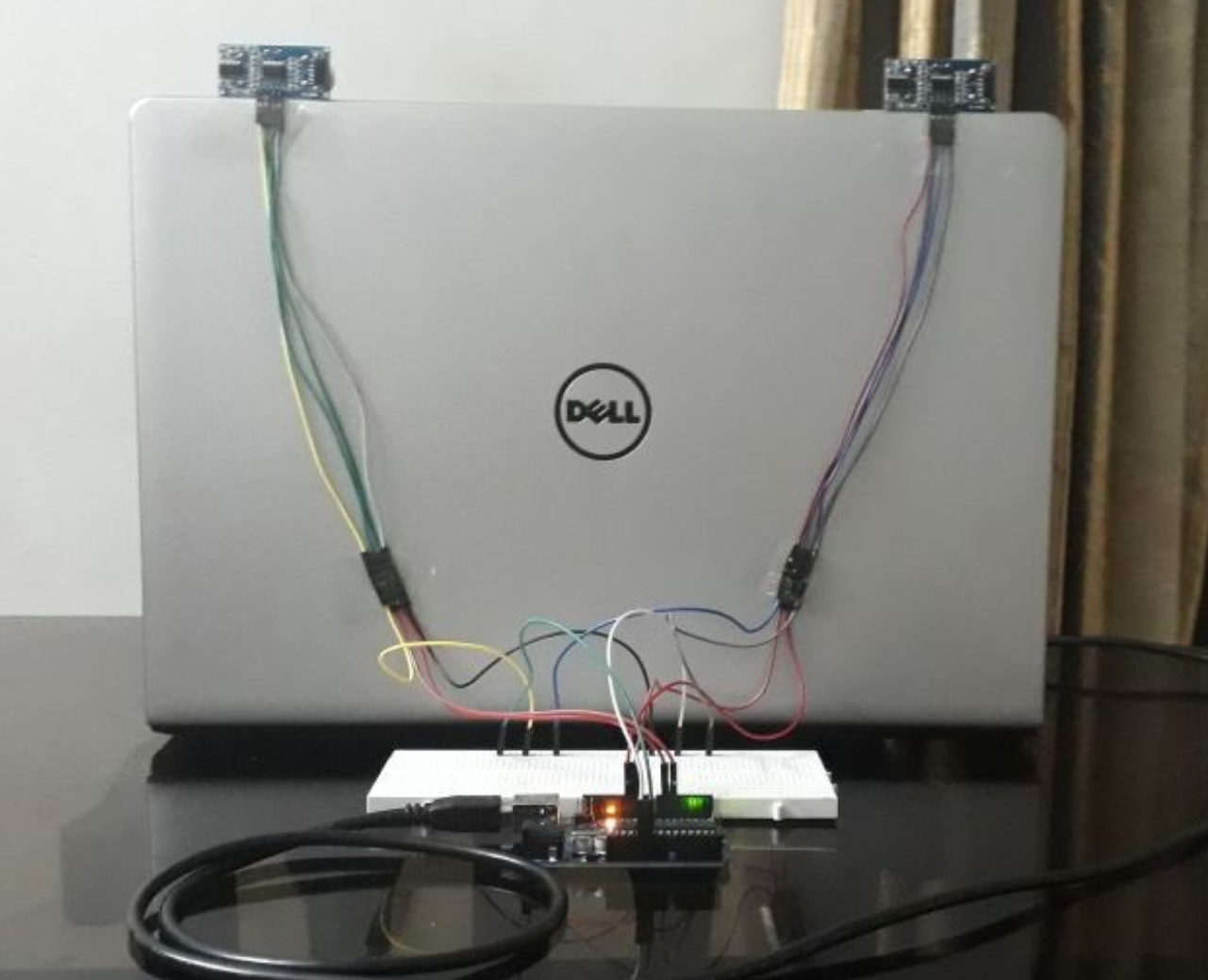


|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Pin Name** | **Description** |
| 1 | Vcc | The Vcc pin powers the sensor, typically with +5V |
| 2 | Trigger | Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave. |
| 3 | Echo | Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor. |
| 4 | Ground | This pin is connected to the Ground of the system. |

### **HC-SR04 Sensor Features**

* Operating voltage: +5V
* Theoretical  Measuring Distance: 2cm to 450cm
* Practical Measuring Distance: 2cm to 80cm
* Accuracy: 3mm
* Measuring angle covered: <15°
* Operating Current: <15mA
* Operating Frequency: 40Hz

****

****

1. **SOFTWARE IMPLEMENTATION**

#### Arduino Code

|  |  |
| --- | --- |
|  | const int trigPin1 = 11; // the number of the trigger output pin ( sensor 1 ) |
|  | const int echoPin1 = 10; // the number of the echo input pin ( sensor 1 ) |
|  | const int trigPin2 = 6; // the number of the trigger output pin ( sensor 2 ) |
|  | const int echoPin2 = 5; // the number of the echo input pin ( sensor 2 ) |
|  |  |
|  | ////////////////////////////////// variables used for distance calculation |
|  | long duration; |
|  | int distance1, distance2; |
|  | float r; |
|  | unsigned long temp=0; |
|  | int temp1=0; |
|  | int l=0; |
|  | //////////////////////////////// |
|  |  |
|  | void find\_distance (void); |
|  |  |
|  | // this function returns the value in cm. |
|  | /\*we should not trigger the both ultrasonic sensor at the same time. |
|  | it might cause error result due to the intraction of the both soundswaves.\*/ |
|  | void find\_distance (void) |
|  | { |
|  | digitalWrite(trigPin1, LOW); |
|  | delayMicroseconds(2); |
|  | digitalWrite(trigPin1, HIGH); |
|  | delayMicroseconds(10); |
|  | digitalWrite(trigPin1, LOW); |
|  |  |
|  | duration = pulseIn(echoPin1, HIGH, 5000);// here this pulsein function wont wait more then 5000us for the ultrasonic sound to came back. (due to this it wont measure more than 60cm) |
|  | // it helps this project to use the gesture control in the defined space. |
|  | // so that, it will return zero if distance greater then 60m. ( it helps usually if we remove our hands infront of the sensors ). |
|  |  |
|  | r = 3.4 \* duration / 2; // calculation to get the measurement in cm using the time returned by the pulsein function. |
|  | distance1 = r / 100.00; |
|  | /////////////////////////////////////////upper part for left sensor and lower part for right sensor |
|  | digitalWrite(trigPin2, LOW); |
|  | delayMicroseconds(2); |
|  | digitalWrite(trigPin2, HIGH); |
|  | delayMicroseconds(10); |
|  | digitalWrite(trigPin2, LOW); |
|  |  |
|  | duration = pulseIn(echoPin2, HIGH, 5000); |
|  | r = 3.4 \* duration / 2; |
|  | distance2 = r / 100.00; |
|  | delay(100); |
|  | } |
|  |  |
|  | void setup() |
|  | { |
|  | Serial.begin(9600); |
|  | pinMode(trigPin1, OUTPUT); // initialize the trigger and echo pins of both the sensor as input and output: |
|  | pinMode(echoPin1, INPUT); |
|  | pinMode(trigPin2, OUTPUT); |
|  | pinMode(echoPin2, INPUT); |
|  | delay (1000); |
|  |  |
|  | } |
|  |  |
|  | void loop() |
|  | { |
|  | find\_distance(); |
|  |  |
|  |  |
|  | if(distance2<=35 && distance2>=15) |
|  | { |
|  | temp=millis(); |
|  | while(millis()<=(temp+300)) |
|  | find\_distance(); |
|  | if(distance2<=35 && distance2>=15) |
|  | { |
|  | temp=distance2; |
|  | while(distance2<=50 || distance2==0) |
|  | { |
|  | find\_distance(); |
|  | if((temp+6)<distance2) |
|  | { |
|  | Serial.println("down"); |
|  | } |
|  | else if((temp-6)>distance2) |
|  | { |
|  | Serial.println("up"); |
|  | } |
|  | } |
|  | } |
|  | else |
|  | { |
|  | Serial.println("next"); |
|  | } |
|  | } |
|  |  |
|  | else if(distance1<=35 && distance1>=15) |
|  | { |
|  |  |
|  | temp=millis(); |
|  |  |
|  | while(millis()<=(temp+300)) |
|  | { |
|  | find\_distance(); |
|  | if(distance2<=35 && distance2>=15) |
|  | { |
|  | Serial.println("change"); |
|  | l=1; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | if(l==0) |
|  | { |
|  | Serial.println("previous"); |
|  | while(distance1<=35 && distance1>=15) |
|  | find\_distance(); |
|  | } |
|  | l=0; |
|  | } |
|  |  |
|  | } |

**Python Code**

|  |  |
| --- | --- |
|  | import serial |
|  | import pyautogui |
|  |  |
|  | Arduino\_Serial = serial.Serial('com12',9600) |
|  |  |
|  | while 1: |
|  | incoming\_data = str (Arduino\_Serial.readline()) |
|  | print incoming\_data |
|  |  |
|  |  |
|  | if 'next' in incoming\_data: |
|  | pyautogui.hotkey('ctrl', 'pgdn') |
|  |  |
|  | if 'previous' in incoming\_data: |
|  | pyautogui.hotkey('ctrl', 'pgup') |
|  |  |
|  | if 'down' in incoming\_data: |
|  | #pyautogui.press('down') |
|  | pyautogui.scroll(-100) |
|  |  |
|  | if 'up' in incoming\_data: |
|  | #pyautogui.press('up') |
|  | pyautogui.scroll(100) |
|  |  |
|  | if 'change' in incoming\_data: |
|  | pyautogui.keyDown('alt') |
|  | pyautogui.press('tab') |
|  | pyautogui.keyUp('alt') |
|  |  |
|  | incoming\_data = ""; |

**4. CONCLUSION AND FUTURE WORK**

**4.1 CONCLUSION**

* The arduino based hand gesture control of your laptop was built and implemented.
* The system is targeted at younger generation.
* The prototype developed can control electrical your laptop and desktops.
* The preliminary test results are promising.

**4.2 FUTURE WORK**

* The above project can also be implemented for advanced gaming which include motion-based control, hand gestures.
* The above project can also be utilized to perform or control various task with respect to watching movie for example, play or pause increase or decrease the volume.
* Gesture control has been widely used in the rising field of Virtual Reality. VR headsets are widely used with computer games but they are also used in other applications, including simulators and trainers. They comprise a stereoscopic head-mounted display (providing separate images for each eye), stereo sound, and head motion tracking sensors (which may include gyroscopes, accelerometers, structured light systems, etc.). Some VR headsets also have eye tracking sensors and gaming controllers.

**5. REFERENCES**

1. https://www.arduino.cc/
2. <https://www.electronicshub.org/arduino-based-hand-gesture-control-computer/>
3. <https://playground.arduino.cc/Interfacing/Python>
4. <http://forefront.io/a/beginners-guide-to-arduino>
5. [www.circuitstoday.com](http://www.circuitstoday.com/)
6. [www.wikipedia.com](http://www.wikipedia.com/)