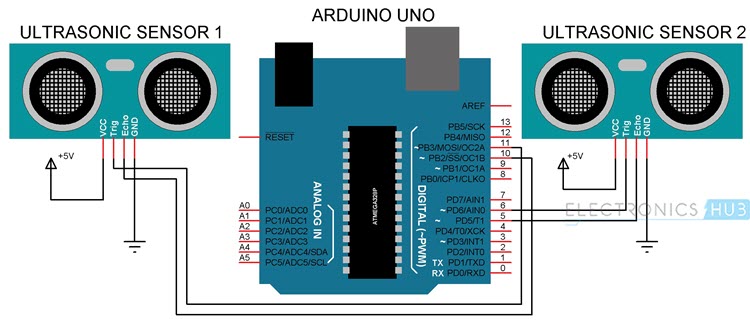
PROJECT

Arduino based Hand Gesture Control of Computer

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.

Circuit and Working



PIN CONNECTIONS

trigger Pin = 11 ( sensor 1 )

echo Pin = 10 ( sensor 1 )

trig Pin = 6 ( sensor 2 )

echoPin2 = 5 ( sensor 2 )

vcc to 5v in arduino

gnd to gnd arduino

WORKING

The following are the 5 different hand gestures or actions that I’ve programmed for demonstration purpose.

**Gesture 1:** Place your hand in front of the Right Ultrasonic Sensor at a distance (between 15CM to 35CM) for a small duration and move your hand away from the sensor. This gesture will Scroll Down the Web Page or Decrease the Volume.

**Gesture 2:** Place your hand in front of the Right Ultrasonic Sensor at a distance (between 15CM to 35CM) for a small duration and move your hand towards the sensor. This gesture will Scroll up the Web Page or Increase the Volume.

**Gesture 3:** Swipe your hand in front of the Right Ultrasonic Sensor. This gesture will move to the Next Tab.

**Gesture 4:** Swipe your hand in front of the Left Ultrasonic Sensor. This gesture will move to the Previous Tab or Play/Pause the Video.

**Gesture 5:** Swipe your hand across both the sensors (Left Sensor first). This action will Switch between Tasks.

CODE for arduino

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(); // this function will stores the current distance measured by the ultrasonic sensor in the global variable "distance1 and distance2"

// no matter what, the program has to call this "find\_distance" function continuously to get the distance value at all time.

if(distance2<=35 && distance2>=15) // once if we placed our hands in front of the right sensor in the range between 15 to 35cm this condition becomes true.

{

temp=millis(); // store the current time in the variable temp. (" millis " Returns the number of milliseconds since the Arduino board began running the current program )

while(millis()<=(temp+300)) // this loop measures the distance for another 300 milliseconds. ( it helps to find the difference between the swipe and stay of our hand in front of the right sensor )

find\_distance();

if(distance2<=35 && distance2>=15) // this condition will true if we place our hand in front of the right sensor for more then 300 milli seconds.

{

temp=distance2; // store the current position of our hand in the variable temp.

while(distance2<=50 || distance2==0) // this loop will run untill we removes our hand in front of the right sensor.

{

find\_distance(); // call this function continuously to get the live data.

if((temp+6)<distance2) // this condition becomes true if we moves our hand away from the right sensor (\*\*but in front of it ). here " temp+6 " is for calibration.

{

Serial.println("down"); // send "down" serially.

}

else if((temp-6)>distance2) // this condition becomes true if we moves our hand closer to the right sensor.

{

Serial.println("up"); // send "up" serially.

}

}

}

else // this condition becomes true, if we only swipe in front of the right sensor .

{

Serial.println("next"); // send "next" serially.

}

}

else if(distance1<=35 && distance1>=15) // once if we placed our hands in front of the left sensor in the range between 15 to 35cm this condition becomes true.

{

temp=millis();

while(millis()<=(temp+300))

{

find\_distance();

if(distance2<=35 && distance2>=15) // if our hand detects in the right sensor before 300 milli seconds this condition becomes true. ( usually it happens if we swipe our hand from left to right sensor )

{

Serial.println("change"); // send "change" serially.

l=1; // store 1 in variable l. ( it avoids the program to enter into the upcoming if condition )

break; // break the loop.

}

}

if(l==0) // this condition will become true, only if we swipe our hand in front of left sensor.

{

Serial.println("previous"); // send "previous" serially.

while(distance1<=35 && distance1>=15) // this loop will rotate untill we removes our hand infront of the left sensor. this will avoid not to enter this if condition again.

find\_distance();

}

l=0; // make l=0 for the next round.

}

}

CODE for python IDE (2.7)

import serial # add Serial library for serial communication

import pyautogui # add pyautogui library for programmatically controlling the mouse and keyboard.

Arduino\_Serial = serial.Serial('com8',9600) # Initialize serial and Create Serial port object called Arduino\_Serial

while 1:

incoming\_data = str (Arduino\_Serial.readline()) # read the serial data and print it as line

print incoming\_data # print the incoming Serial data

if 'next' in incoming\_data: # if incoming data is 'next'

pyautogui.hotkey('ctrl', 'pgdn') # perform "ctrl+pgdn" operation which moves to the next tab

if 'previous' in incoming\_data: # if incoming data is 'previous'

pyautogui.hotkey('ctrl', 'pgup') # perform "ctrl+pgup" operation which moves to the previous tab

if 'down' in incoming\_data: # if incoming data is 'down'

#pyautogui.press('down') # performs "down arrow" operation which scrolls down the page

pyautogui.scroll(-100)

if 'up' in incoming\_data: # if incoming data is 'up'

#pyautogui.press('up') # performs "up arrow" operation which scrolls up the page

pyautogui.scroll(100)

if 'change' in incoming\_data: # if incoming data is 'change'

pyautogui.keyDown('alt') # performs "alt+tab" operation which switches the tab

pyautogui.press('tab')

pyautogui.keyUp('alt')

incoming\_data = ""; # clears the data

