CONTROLAREA UNUI LAPTOP FOLOSIND ARDUINO

Dragu Rebeca
2.2

PUNCTE CHEIE

- **Diagrama circuitului**
- **Designul proiectului**
- **Arduino**
- **Codul Python**

INTRODUCERE

Proiectul Arduino, bazat pe controlarea unui laptop cu senzori ultrasonici, este implementat folosind Python.

În acest proiect se pot găsi elementele de bază despre cum să utilizaţi o placă de dezvoltare Arduino cu Python, instalarea Python pe computer, configurarea Bibliotecii Serial (important pentru comunicarea cu Arduino) şi codurile de proiect.

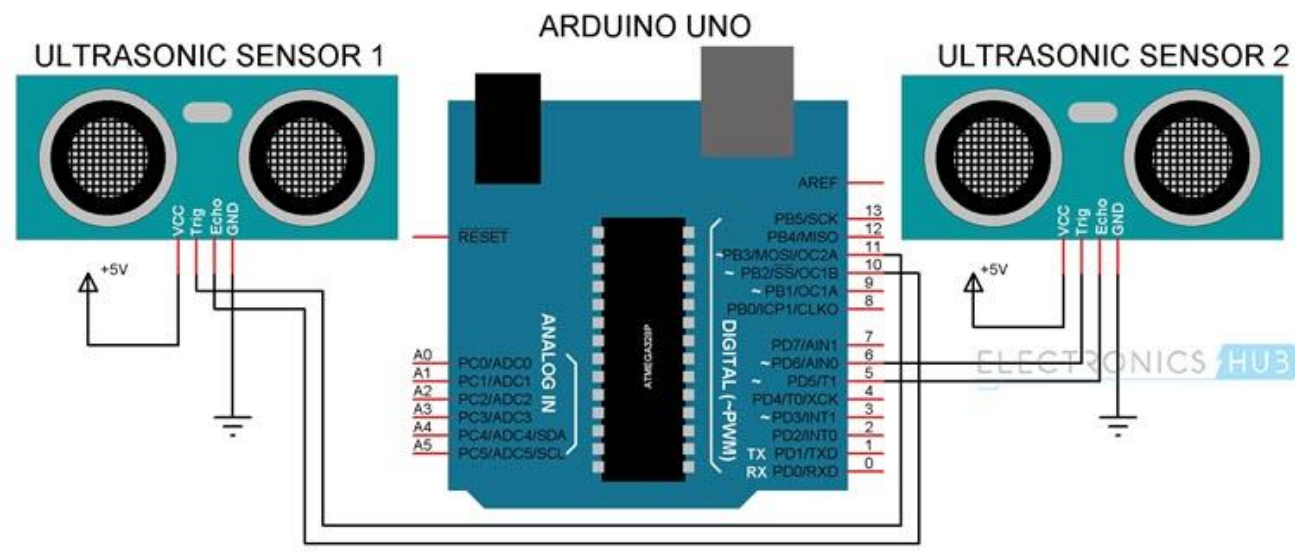
Principiul din spatele controlului prin gesturi ale mâinii, bazat pe Arduino, este de fapt foarte simplu. Tot ce trebuie făcut este să folosim doi senzori ultrasonici cu Arduino, aşezarea mâinii în faţa senzorului şi calcularea distantei dintre mână şi senzor. Folosind aceste informaţii, pot fi efectuate acţiuni relevante în computer.

Am aşezat cei doi senzori în partea de sus a ecranului unui laptop. Informaţiile despre distanţă de la Arduino sunt colectate de un program Python şi o bibliotecă specială numită PyAutoGUI, care va converti datele în acţiuni de clic de la tastatură.

DIAGRAMA CIRCUITULUI

Ce trebuie să facem este să folosim doi senzori ultrasonici și Arduino, să așezăm mâna în fața senzorului și acesta o să calculeze distanța dintre mână și senzor. Folosind aceste informații, pot fi efectuate acțiuni relevante în computer.

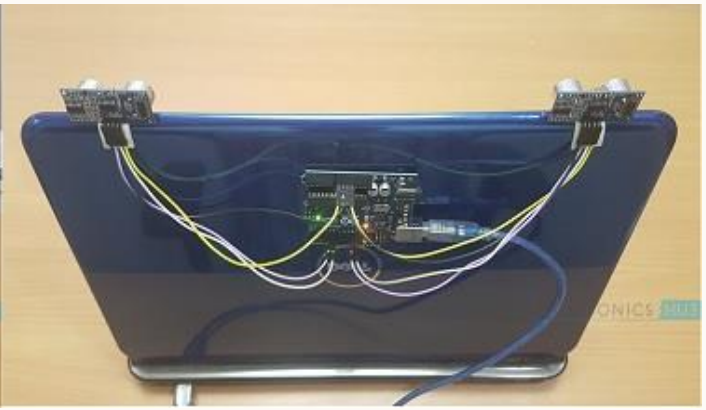
Schema de circuit a unei părți a proiectului Arduino este prezentată în imaginea următoare. Este format dintr-o placă Arduino UNO și doi senzori HC-SR04 și putem alimenta toate componentele de la portul USB al laptopului.



DESIGNUL PROIECTULUI

Pinii de declanșare și eco ai primului senzor (care este plasat în partea stângă a ecranului) sunt conectați la Pinii 11 și 10 ai Arduino. Pentru al doilea senzor , pinii de declanșare și eco sunt conectați la pinii 6 și 5 ai Arduino.

Acum, venind la amplasarea senzorilor, am plasat ambii senzori deasupra ecranului laptopului, unul la capătul din stânga și celălalt la dreapta.



Gestul 1: Așezați mâna în fața senzorului ultrasonic drept la o distanță (între 10 cm și 25 cm) pentru o perioadă mică și îndepărtați-vă mâna de senzor. Acest gest va derula în jos pagina web sau va reduce volumul.
Gestul 2: Așezați mâna în fața senzorului drept la o distanță (între 10 cm și 25 cm) pentru o perioadă mică și mutați-vă mâna spre senzor. Acest gest va derula în sus pagina web sau va crește volumul.
Gestul 3: Glisați mâna în fața senzorului drept. Prin acest gest se va realiza mutarea la pagina următoare.
Gestul 4: Glisați mâna în fața senzorului cu ultrasonic din stânga. Prin acest gest se va realiza mutarea la pagina următoare.

CODUL ARDUINO

```
/*
 * gesture control program for controlling certain functions in windows pc
 * Code by BalaAppu
 * Website: www.electronicshub.org
 */

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.
   }
}
```

CODUL ÎN PYTHON

```
import serial                                   # add Serial library for serial communication
import pyautogui                                # add pyautogui library for programmatically controlling the mouse and keyboard.

Arduino_Serial = serial.Serial('com12',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
```

MULȚUMESC PENTRU ATENȚIA ACORDATĂ !

Dragu Rebeca-Alina
AC, an 2 IS, subgrupa 2.2