Hand Gesture Control pentru Computer

Autor: Raliță Andreea-Mădălina
Grupa: 335CA

Introducere

Printr-o îmbinare de cod în Arduino cu un modul în Python, acest proiect oferă o modalitate creativă de a manipula laptop-ul, fără a atinge tastatura sau mouse-ul. Astfel, putem naviga prin tab-uri sau controla funcționalitățile de redare a unui videoclip într-un mod simplu, ușor și distractiv.

Descriere generală

Din punctul de vedere al componentelor, proiectul este unul destul de simplist. Prin intermediul a doi senzori, voi evalua și calcula distanța și direcția în care are loc mișcarea mâinilor, iar pentru fiecare tip de mișcare am atribuit o funcționalitate, după cum urmează:

- **Gestul 1** → Ne poziționăm mâna la o distanță mică față de senzorul stâng, cuprinsă între 15 și 35 cm, și o coborâm ușor înspre el. Acest gest va mări volumul videoclipului redat sau va naviga în susul paginii deschise.
- **Gestul 2** → Cu mâna poziționată ca mai devreme, o îndepărtăm ușor de senzorul stâng, mergând cu ea în sus. Acest gest va diminua volumul videoclipului redat sau va naviga în josul paginii deschise.
- **Gestul 3** → Ne poziționăm mâna la o distanță mică față de senzorul drept, cuprinsă între 15 și 35 cm, și o coborâm ușor înspre el. Acest gest va face Rewind pentru videoclipul redat, adică va sări peste câteva secunde față de momentul curent în care ne aflăm.
- **Gestul 4** → Cu mâna poziționată ca mai devreme, o îndepărtăm ușor de senzorul drept, mergând cu ea în sus. Acest gest va face Forward pentru videoclipul redat, adică ne va duce câteva secunde înapoi.
- **Gestul 5** → Plimbăm scurt mâna pe deasupra senzorului stâng (swipe) pentru a trece la tab-ul din stânga tab-ului curent.
- **Gestul 6** → Plimbăm scurt mâna pe deasupra senzorului drept (swipe) pentru a trece la tab-ul din dreapta tab-ului curent.

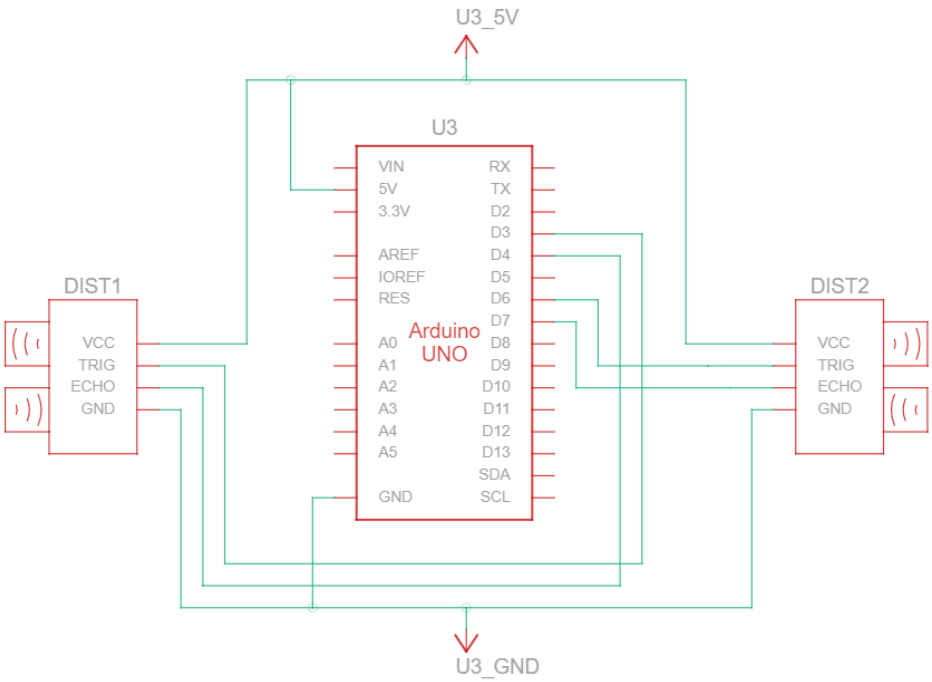
Schemă bloc

Hardware Design

Lista componente

Nume	Număr Piese
Arduino UNO	1
Ultrasonic Sensor	2
Breadboard	1
Fire mama-tata	8
Fire tata-tata	1
Laptop cu conexiune la internet	1

Schemă electrică



Software Design

Mediul de dezvoltare:

- software:
  - Arduino IDE [1)]
  - PyAutoGUI [2)]
- pentru realizarea schemei-bloc:

    - draw.io [3)]
- pentru realizarea schemei electrice:
    - EAGLE [4)]

Biblioteci folosite:

- pentru comunicarea dintre Python si Arduino:
    - Serial library
- pentru a converti datele in functionalitati ale tastaturii:
    - PyAutoGUI

Structura codului

Codul este stucturat în majoritatea lui în Arduino IDE, iar codul Arduino generează 6 comenzi care sunt trimise mai departe portului serial. Folosind aceste 6 comenzi, am scris un modul în Python care să manipuleze comenzile primite, invocând o anume combinație de taste pentru fiecare comandă în parte.

Arduino IDE - codul Arduino include următoarele funcții:

- *calculate_distance(trigger, echo)* → calculează în centimetri distanța la care se află mâna de senzor, folosindu-se de funcția pulseIn pentru a determina durata pulsului HIGH, după care se aplică o formulă pentru a obține distanța în centimetri. Această funcție va fi folosită pentru a calcula distanța atât pentru senzorul stâng, cât și pentru senzorul drept, în funcție de parametrii pe care îi primește.

```
// Calculation to get the measurement of the distance in cm
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;
}
```

- *setup()* → inițializez pinii de trigger și echo pentru fiecare senzor ca input și ca output.

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

  // Initialize the trigger and echo pins of both sensors as input and output
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);
}
```

- *loop()* → colecție de calcule și comparații pentru a identifica comanda sub care se plasează mișcarea efectuată. Pentru fiecare comandă, calculăm distanța până la senzorul corespunzător, și în funcție de încadrarea între anumite limite, stabilim ce comandă vom trimite mai departe portului serial.

Spre exemplu, pentru comenzile “Volume Up”/ “Volume Down”, calculăm distanța de la mână la senzorul stâng, iar dacă ea se află în intervalul [10 cm, 20 cm], continuăm să calculăm distanța în fiecare moment, pentru a determina dacă mâna se apropie (caz în care trimitem “Vup” pe seriala) sau se îndepărtează (vom trimite “Vdown”).

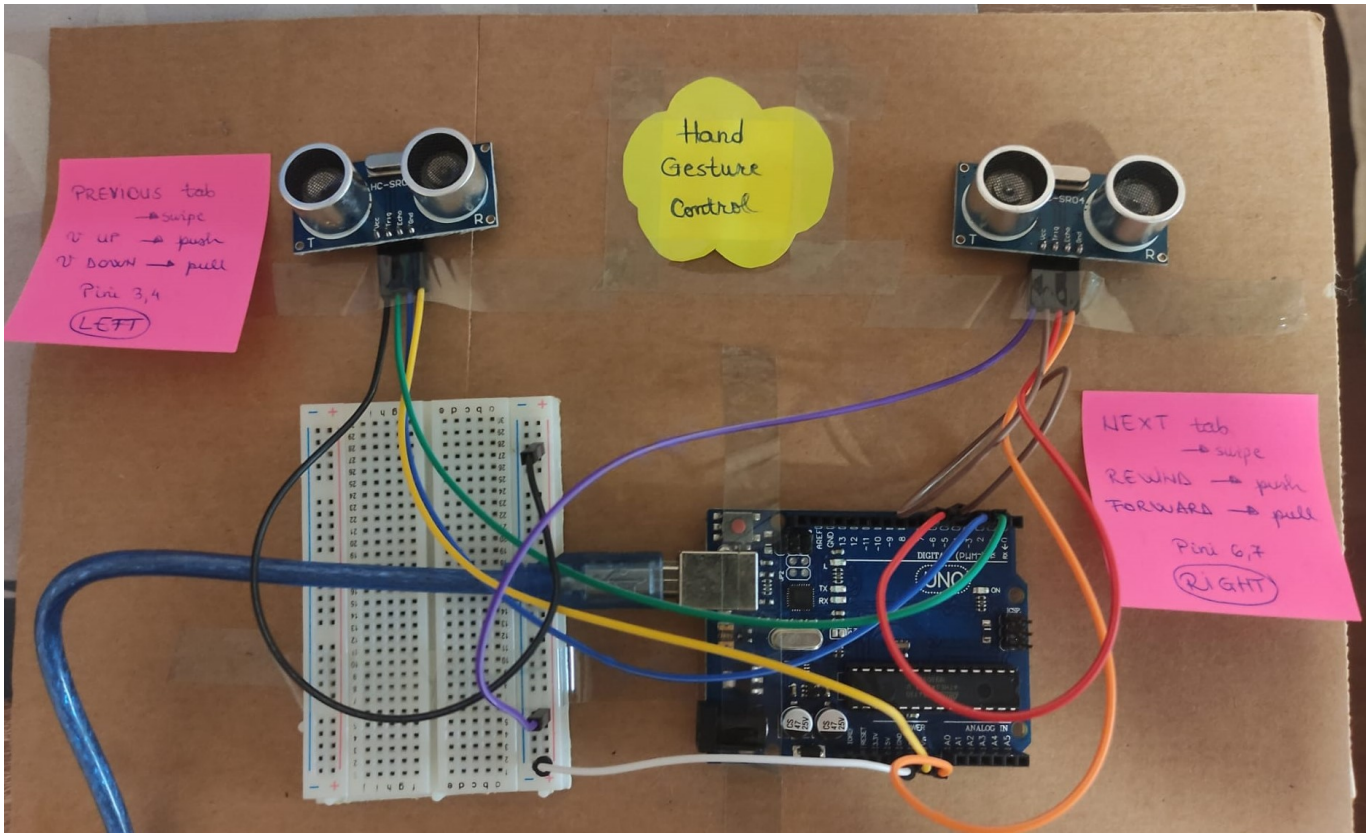
```
// "Vup" and "Vdown" commands
calculate_distance(trigPin1,echoPin1);
distL = dist;

if (distL >= 10 && distL <= 20)
{
  delay(50); //Hand Hold Time
  calculate_distance(trigPin1,echoPin1);
  distL = dist;

  if (distL >= 10 && distL <= 20)
  {
    Serial.println("Left Locked");
    while(distL <= 40)
    {
      calculate_distance(trigPin1,echoPin1);
      distL = dist;

      if (distL < 15) //Hand pushed in
      {
          Serial.println ("Vup");
          delay (300);}
      if (distL > 20) //Hand pulled out
      {
         Serial.println ("Vdown");
         delay (300);}
    }
  }
}
```

Modul Python - codul în Python cuprinde următoarele funcționalități:

- *Arduino_Serial = serial.Serial('com3',9600)* → inițializarea serialei și crearea obiectului de tip Serial port numit Arduino_Serial.
- *incoming_data = str (Arduino_Serial.readline())* → citirea datelor de pe serială și afișarea lor pe o linie.
- *valorile pe care le poate lua incoming_data și combinațiile de taste pe care acestea le invocă* → folosesc funcțiile puse la dispoziție de biblioteca PyAutoGUI pentru manipularea tastaturii, funcții precum hotkey() și press().

```
if 'next' in incoming_data:
    pyautogui.hotkey('ctrl', 'tab')

if 'previous' in incoming_data:
    pyautogui.hotkey('ctrl', 'shift', 'tab')

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

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

if 'Vup' in incoming_data:
    pyautogui.press('up')

if 'Vdown' in incoming_data:
    pyautogui.press('down')
```

Rezultate Obţinute



Concluzii

Proiectul “Hand Gesture Control” este un proiect de dificultate mică spre medie, care însă mi-a plăcut prin funcționalitatea pe care o oferă și prin faptul că mi-a demonstrat că nu trebuie neapărat să folosesc Arduino IDE de sine stătător, ci că pot obtine rezultate încântătoare combinându-l și cu alte

medii de lucru, cum ar fi Python. Menționez că inițial aveam mai multe gesturi pe care senzorii le recunoșteau (din punct de vedere al codului), însă în practică nu mai făceau diferența între ele deoarece erau foarte asemănătoare. Așa că am decis să păstrez mai puține, însă ele să fie recunoscute rapid de senzori.

Download

Arhiva ce conține codul sursă: hand_gesture_control_ralita_andreea_335ca.zip

Pentru a rula proiectul, deschidem fișierul hand_gesture_python.py în Python, apăsăm pe Run Module, după care putem începe să gesticulăm pe deasupra senzorilor. Vom observa că pe măsură ce acțiunile noastre sunt efectuate pe laptop, numele acțiunii va fi și printat explicit ca output.

Bibliografie/Resurse

Instalarea de biblioteci si tool-uri necesare pentru a combina Arduino cu Python:
https://www.electronicshub.org/controlling-arduino-led-python/

Documentatie pentru PyAutoGUI Keyboard Control Functions:
https://pyautogui.readthedocs.io/en/latest/keyboard.html

Export to PDF

[1] https://www.arduino.cc/en/software
[2] https://pyautogui.readthedocs.io/en/latest
[3] https://app.diagrams.net
[4] https://www.autodesk.com/products/eagle/overview?term=1-YEAR&tab=subscription