CubeMaster

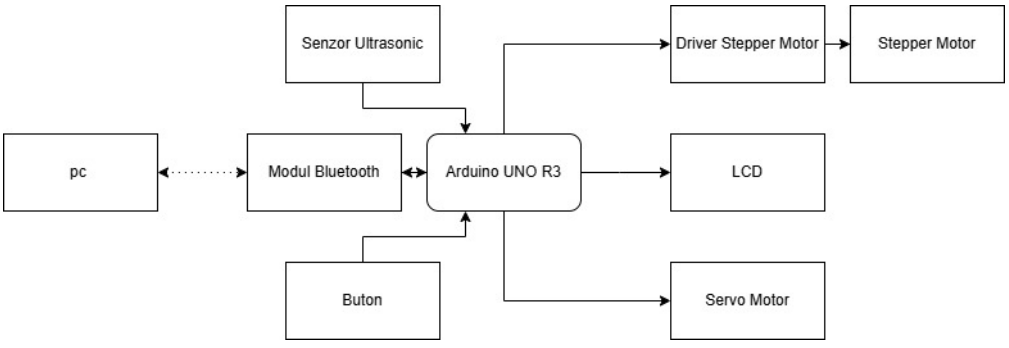
Introducere

CubeMaster este un sistem automat de rezolvare al unui cub rubik ce va consta intr-un sistem de inregistrare al fetelor cubului, generarea unui numar optim de pasi pentru generarea unei solutii, si manipularea cubului prin intermediul unui mecanism simpu de gestionare al rotiriii celor 6 fete pentru rezolvarea acestuia

Descriere generală

CubeMaster este un sistem automat de rezolvare al unui cub Rubik. Scopul său este de a rezolva cubul în cel mai scurt timp posibil, eliminând necesitatea unei intervenții umane pentru rezolvare. Ideea a pornit de la dorința de a automatiza procesul de rezolvare a Cubului Rubik și de a demonstra abilitățile tehnice în domeniul programării și ingineriei electrice. Acest proiect este util atât pentru pasionații de Cub Rubik, care vor aprecia eficiența și precizia sistemului, cât și pentru cei interesați de dezvoltarea de sisteme de control automat și roboți.

Schema bloc



Descriere Module:
PC: Realizeaza preprocesarea necesara rezolvarii cubului: interfata grafica pentru inputul fetelor, rezolvarea cubului (in miscari U F R L B D), translatarea in miscari posibile cu 2 motoare (D, I, J) si transmiterea seriala catre modulul bluetooth.
Modul Bluetooth: Realizeaza conexiunea intre PC si arduino prin protocolul bluetooth, realizeaza transferul digital de date.
Senzor Ultrasonic: Realizeaza detectarea prezentei cubului pe dispozitiv cu rol in pornirea sitemului de rezolvare al cubului sau a functiei de shuffle.
Buton: Facilitează interacțiunea dintre utilizator și dispozitiv pentru alegerea starii dorite, rezolvarea

sau amestecarea cubului.
Driver Stepper Motor + Stepper Motor: Detine in totalitate controlul standului pe care este situat cubul rubik si cu ajutorul caruia poate executa rotirea cubului in procesul de efectuare al miscarilor D, J.
LCD: Afisarea numarului ramas de miscari + configuratia actuala primita de la interfata grafica.
Servo Motor: Detine in totalitate controlul bratului care poate executa rotirea cubului in procesul de efectuare al miscarilor D, I.

Hardware Design

Listă de piese:

PC:
Rol: Preprocesarea. Prin intermediul unei interfate grafice (Vreau sa folosesc camera dar algoritmul de culori nu coopereaza) utilizatorul poate introduce fetele cubului. Cu acestea in memorie se genereaza o solutie si se translateaza in mutari de 6 tipuri pentru fiecare fata (F R L U B D) in miscari de 3 tipuri (D, I, J) apoi se trimite serial la arduino.
D - isi pastreaza scopul, invarte fata inferioara a cubului
I - reprezinta miscarea de rotire al intregului cub dupa cercul F → U → B → D → F unde R si L raman pe loc.
J - reprezinta miscarea de rotire al intregului cub dupa cercul F → L → B → R → F unde U si D raman pe loc.

Dacă ești în căutarea unei metode simple și eficiente de a rezolva Cubul Rubik, acest site e foarte util. Această pagină web detaliază pas cu pas o abordare pentru începători explicand miscarile fundamentale si algoritmii folositi.

Pentru o intelegere mai buna a miscarilor folosite de dispozitiv:
I realizeaza o rotatie a cubului cu 90° in jurul axei X in sensul acelor de ceas.
J realizeaza o rotatie a cubului cu 90° in jurul axei Y in sensul acelor de ceas.

Arduino Uno R3 ATmega328P:
Rol: Unitatea centrala a proiectului. Primeste serial prin modulul bluetooth pasii pentru rezolvarea cubului, la apasarea butonului (si prezenta cubului) daca este o rezolvare incarcata in memorie se porneste mecanismul de rezolvare prin miscarea servo-ului si motorului pas cu pas. In timp ce se rezolva cubul este actualizata si imaginea afisata pe ecranul oled cu numarul de miscari ramase. Conectat la pc prin cablul USB-A ↔ USB-B cu rol strict de alimentare.

HC-05 Master Slave Bluetooth Module with Adapter (3.3 V and 5 V Compatible):
Rol: Comunicarea seriala intre PC si arduino transmisia digitala a mutarilor pentru rezolvarea cubului, e conectat serial cu PC-ul (COM10).
Folosire: Modulul e destul de usor de folosit, plug and play fiind conectat la Rx si Tx pe placa arduino si conectat la PC Pinii folositi: D0(RX) → RXD | D1(TX) → TXD | VCC + GND

Modul Nano micro servo motor SG90:
Rol: Manevrarea bratului pentru a muta cubul Rubik (realizarea miscarilor D, I).
Pinii folositi: D7 → PWM | VCC + GND

Senzor ultrasonic HC-SR04:
Rol: Detectarea prezentei cubului Rubik, oprind executia pana la prezenta acestuia.
Calibrare: Senzorul furnizeaza informatii continuu astfel modificările aduse sunt doar pentru normalizarea valorilor citite.
Pinii folositi: D4 → ECHO | D5 → TRIG | VCC + GND

Motor pas cu angrenaj 28BYJ-48, 12V + Stepper Driver ULN2003 Stepper:
Rol: Manevrarea stand-ului pe care se afla cubul astfel are rol in mutarea cubului Rubik (realizarea miscarilor de D, J).
Pinii folositi: D8 → IN1 | D9 → IN2 | D10 → IN3 | D11 → IN4 | VCC + GND

Placa test breadboard 400:
Element de legatura.

40 x Fire Dupont mama - tata 20 cm:
Elemente de legatura.

40 x Fire Dupont tata-tata 20cm:
Elemente de legatura.

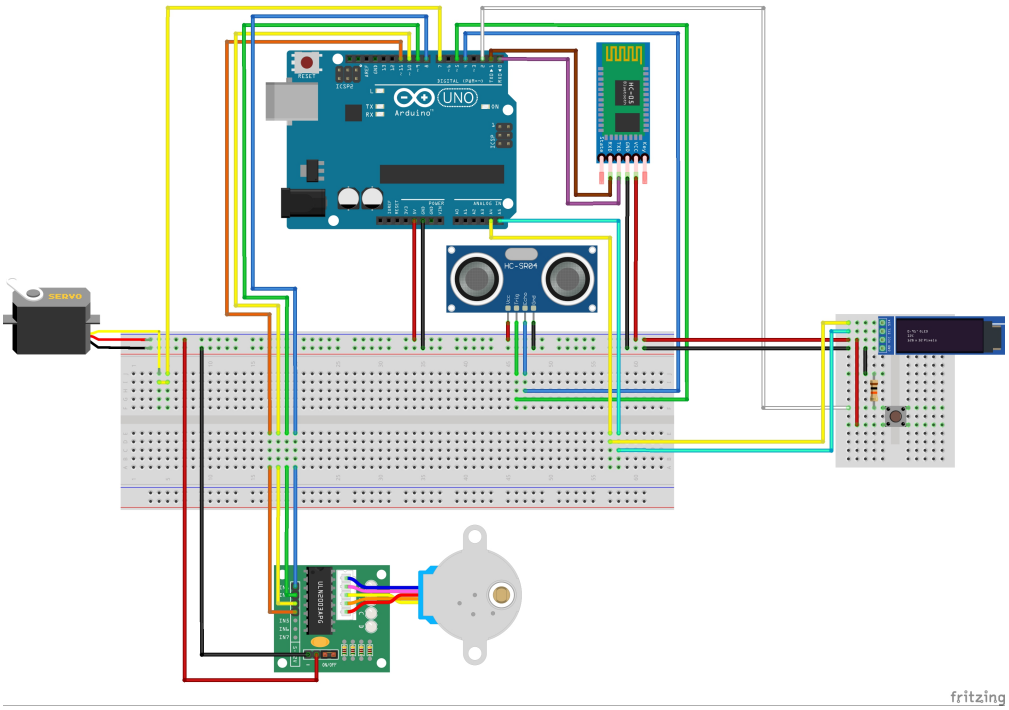
Ecran OLED 0.91”:
Rol: Dispozitiv pentru afisarea numarului de miscari ramase pentru rezolvarea cubului + reprezentarea configuratiei actuale a cubului + mesaje pentru interactiunea dispozitiv - utilizator.
Counterul scade pe masura ce se executa miscari si modifica de asemenea si configuratia.
Driver folosit: SSD1306, I2C
Pinii folositi: A4 → SDA | A5 → SCL | VCC + GND

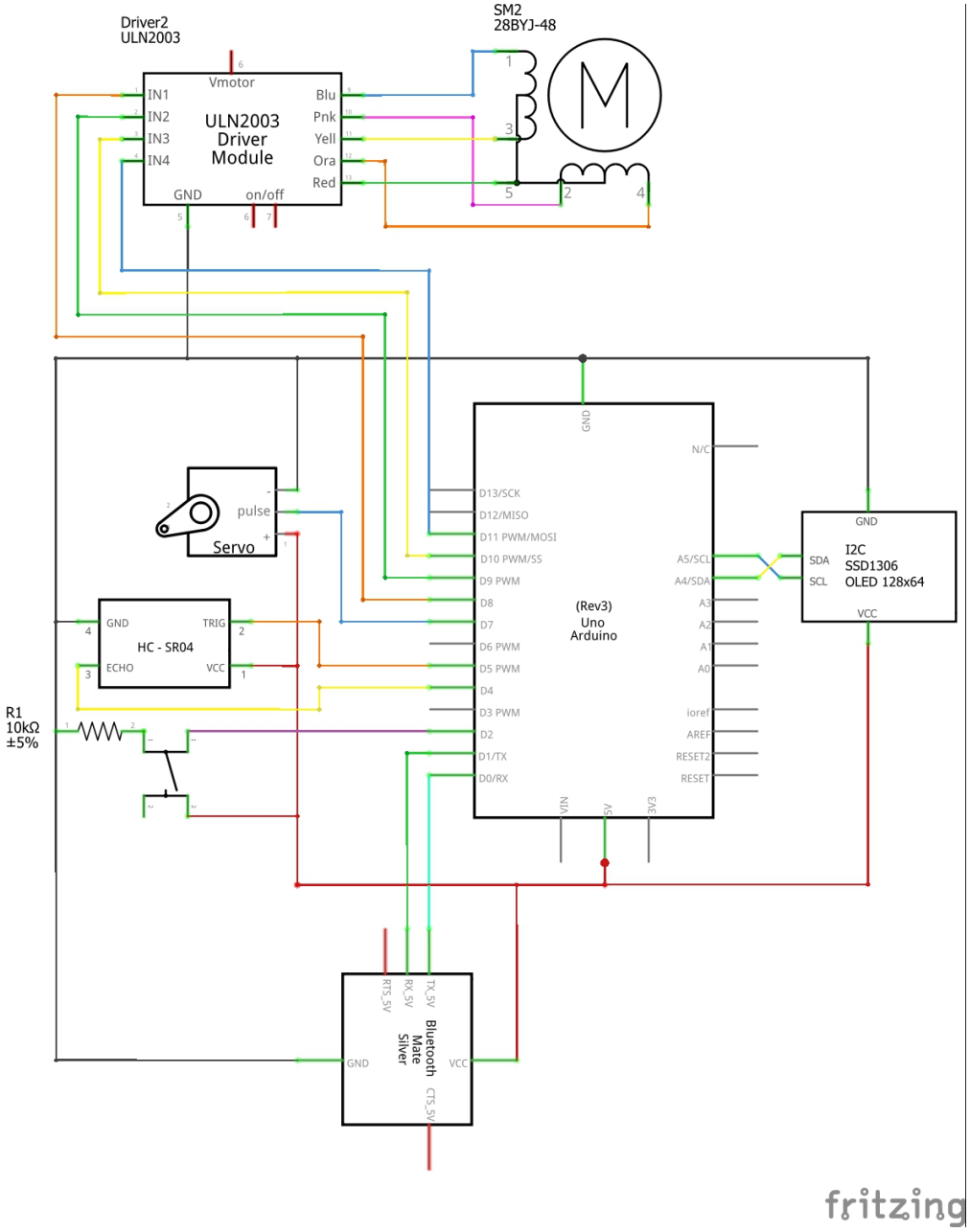
Breadboard 170 puncte:
Element de legatura.

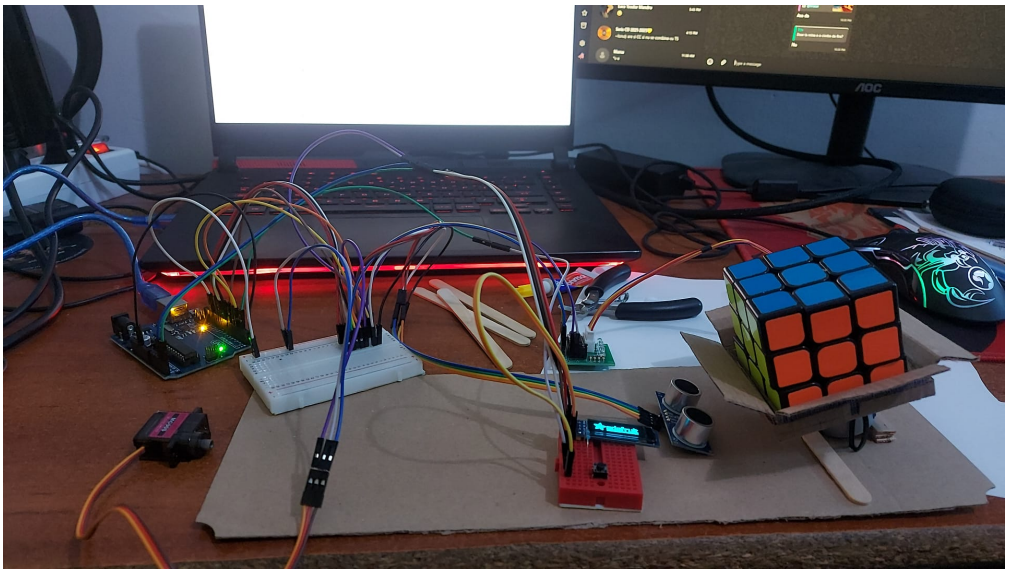
Push button:
Rol: Buton pentru activarea unei rutini de intrerupere, aceasta porneste procesul de amestecare al cubului sau de rezolvare.
Pinii folositi: D2 → Pin pentru intrerupere | VCC + GND

Rezistor:
Rol: Conectat in serie cu butonul (pull down resistor).

Scheme electrice:


SERVO
UNO
Arduino
HC-SR04
fritzing


Driver2
ULN2003
SM2
28BYJ-48
Vmotor
IN1
IN2
IN3
IN4
ULN2003
Driver
Module
GND
on/off
Blu
Pnk
Yell
Ora
Red
M
pulse
Servo
GND
N/C
D13/SCK
D12/MISO
D11 PWM/MOSI
D10 PWM/SS
D9 PWM
D8
D7
D6 PWM
D5 PWM
D4
D3 PWM
D2
D1/TX
D0/RX
(Rev3)
Uno
Arduino
A5/SCL
A4/SDA
A3
A2
A1
A0
ioref
AREF
RESET2
RESET
VIN
5V
3V3
GND
SDA
SCL
I2C
SSD1306
OLED 128x64
VCC
GND
TRIG
HC - SR04
ECHO
VCC
R1
10kΩ
±5%
RTS_5V
RX_5V
TX_5V
Bluetooth
Mate
Silver
GND
VCC
CTS_5V
fritzing



Software Design

Mediu de dezvoltare

În cadrul acestui proiect, am utilizat două medii de dezvoltare distincte: Visual Studio Code pentru partea de Python și Arduino IDE pentru partea de microcontroler. Am ales aceste medii datorită flexibilității și suportului extensiv pentru diverse biblioteci și extensii necesare proiectului si in plus integrarea cu Git pentru inregistrarea progresului software.

Librării şi surse 3rd-party

Visual Studio Code (VSCode)
Pentru acest proiect, am creat un mediu virtual Python în care am instalat următoarele biblioteci esențiale:

1. **~~cv2 (OpenCV):~~** ~~Folosită pentru procesarea imaginilor cu fata cubului pentru a identifica configuratia acestuia.~~
2. **pathlib și Path:** Utilizate pentru manipularea căilor de fișiere de logging in care stochez configuratiile si rezolvarile anterioare.
3. **kociemba:** Folosită pentru rezolvarea cubului Rubik in cel mai eficient mod sub 20 de miscari “God's Number”.
4. **tkinter:** Utilizat pentru realizarea interfației grafice prin intermediul caruia utilizatorului ii este posibil sa insereze configuratia cubului, sa o salveze impreuna cu rezolvarea cubului pe local si posibilitatea de a trimite serial catre CubeMaster pentru a il rezolva automat.
5. **re:** Utilizat pentru expresii regulate in cadrul manipularii configuratiilor si stringurilor de rezolvare.
6. **serial:** Utilizat pentru comunicarea serială cu modulul bluetooth HC-05 din produsul CubeMaster conectat la placa arduino din interior.
7. **time:** Utilizat pentru gestionarea timpului și temporizări.
8. **random:** Utilizat pentru gestionarea randomizarea in cazul functiei de shuffle.

Fun Fact: God's Number
Știai că un cub Rubik poate fi rezolvat în maximum 20 de mișcări? Această valoare este cunoscută sub numele de “God's Number” și reprezintă numărul minim de mișcări necesare pentru a rezolva orice configurație a cubului Rubik. În 2010, cercetătorii au demonstrat că, indiferent de cât de amestecat este un cub Rubik, acesta poate fi întotdeauna adus la starea sa rezolvată în 20 de mișcări sau mai puțin.
Mai multe informatii aici

Arduino IDE

1. **Servo.h:** Pentru controlul bratului robotului care e controlat printr-un servomotor.
2. **Stepper.h:** Pentru controlul standului cubului Rubik care este operat de motorul pas cu pas.
3. **Wire.h:** Pentru comunicarea I2C.
4. **Adafruit_GFX.h și Adafruit_SSD1306.h:** Pentru controlul afișajului OLED.

Surse şi funcţii implementate

Python:
În programul scris în Python, scopul principal este de a genera o interfață grafică pentru utilizator și de a preprocesa datele necesare pentru rezolvarea cubului Rubik. Funcțiile implementate în Python sunt, în mare parte, pentru gestionarea butoanelor din interfața tkinter, comunicarea serială și conversia soluțiilor de mutare standard în mutări compatibile cu brațul controlat de servomotor și stand-ul controlat de motorul pas cu pas. Dintre aceste funcții, cea pentru conversia soluțiilor este cea mai importantă și cea mai complexă.

Generarea Interfeței Grafice cu Tkinter:
Crearea ferestrei principale și configurarea layout-ului pentru afișarea fețelor cubului Rubik.
Implementarea funcției schimba_culoare pentru a permite utilizatorului să modifice culoarea pătratelor printr-un simplu click.
Implementarea funcției salveaza_culori pentru a salva configurația culorilor într-un fișier text și a construi configurația finală pentru rezolvare.
Implementarea funcției rezolva_cub pentru a apela algoritmul de rezolvare kociemba și a converti soluția într-o formă compatibilă cu hardware-ul.

Comunicarea Serială cu Arduino:
Funcția write_read pentru a trimite date către Arduino și a primi răspunsuri.
Funcția send_to_ard_but pentru a trimite soluția calculată la Arduino.
Transformarea Soluțiilor de Mutare:

Preprocesare:
Funcția modify_string pentru a transforma soluțiile de mutare standard într-un format specific utilizat de hardware-ul controlat de servomotor și motorul pas cu pas.

Arduino:
Codul scris pentru Arduino gestionează controlul motoarelor pas cu pas și a servomotorului pentru a efectua mișcările necesare rezolvării cubului Rubik. De asemenea, include funcții pentru afișarea mesajelor pe un ecran OLED, citirea distanței folosind un senzor ultrasonic și gestionarea intrărilor de la un buton.

Inițializarea Componentelor:
Configurarea ecranului OLED, a senzorului ultrasonic, a servomotorului și a motorului pas cu pas.
Atribuirea unui buton pentru a porni secvența de rezolvare.

Funcțiile pentru Rotiri:
Funcții dedicate pentru rotirea fețelor cubului (rotateD, rotateDi, rotateI, rotateJ, rotateJi), fiecare realizând mișcări precise ale motorului pas cu pas și ale servomotorului.

Gestionarea Afisajului OLED:
Funcții pentru afișarea diferitelor mesaje și stări pe ecranul OLED (waittt, doneee, testdrawstyles).

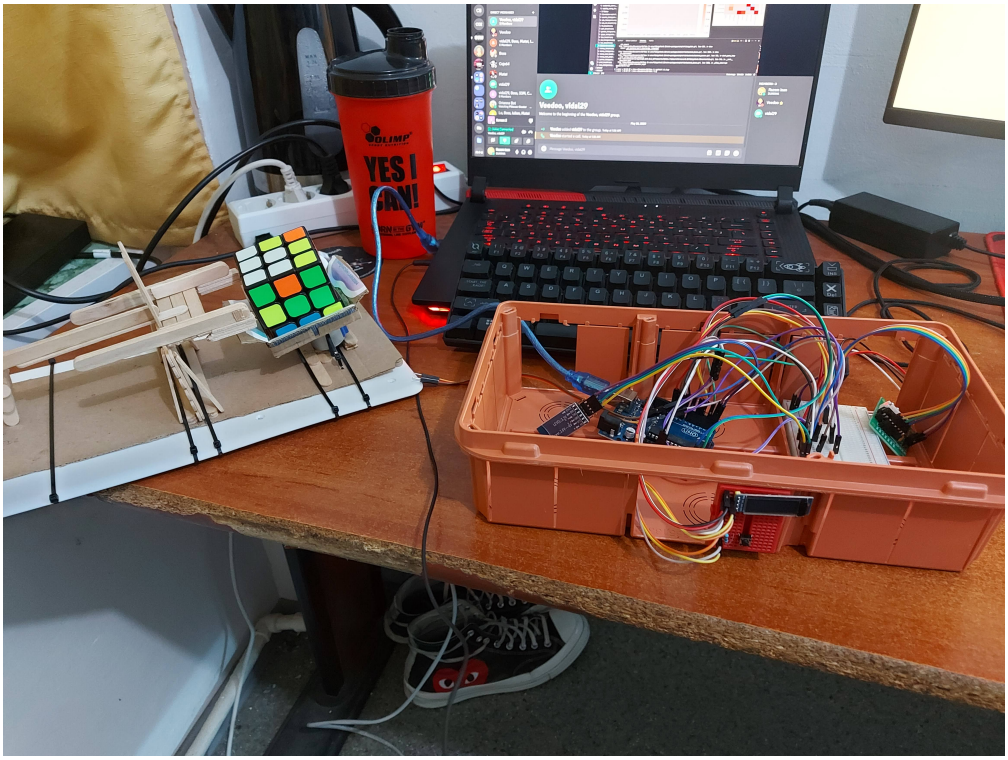
Secvența Principală de Rezolvare:
Funcția loop care așteaptă apăsarea butonului pentru a începe rezolvarea, citește secvența de mișcări de la Python, și apoi efectuează fiecare mișcare în ordine, actualizând ecranul OLED și verificând

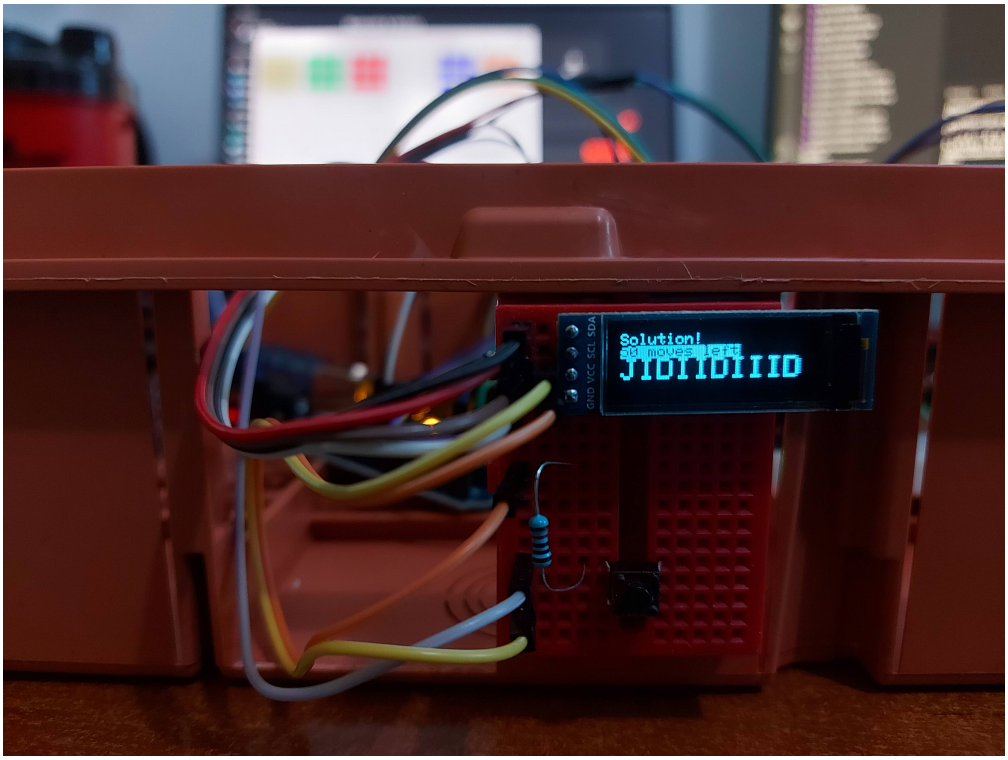
distanța pentru siguranță.

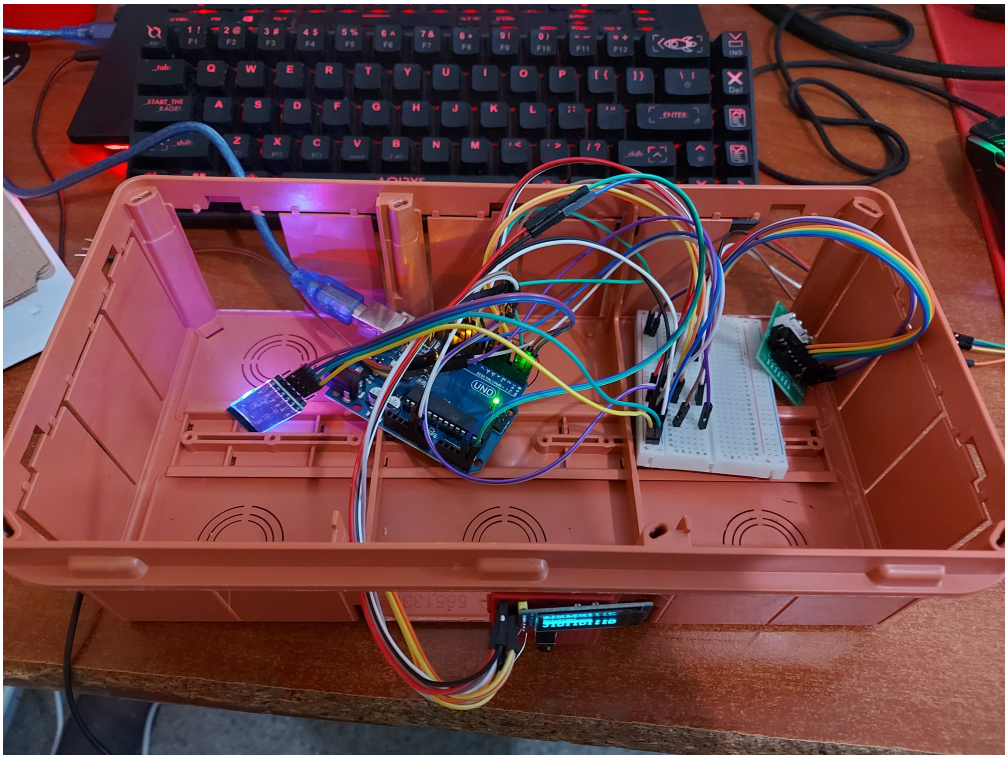
Funcția de Shuffle:
Generarea unei secvențe de mișcări aleatorii pentru amestecarea cubului Rubik.

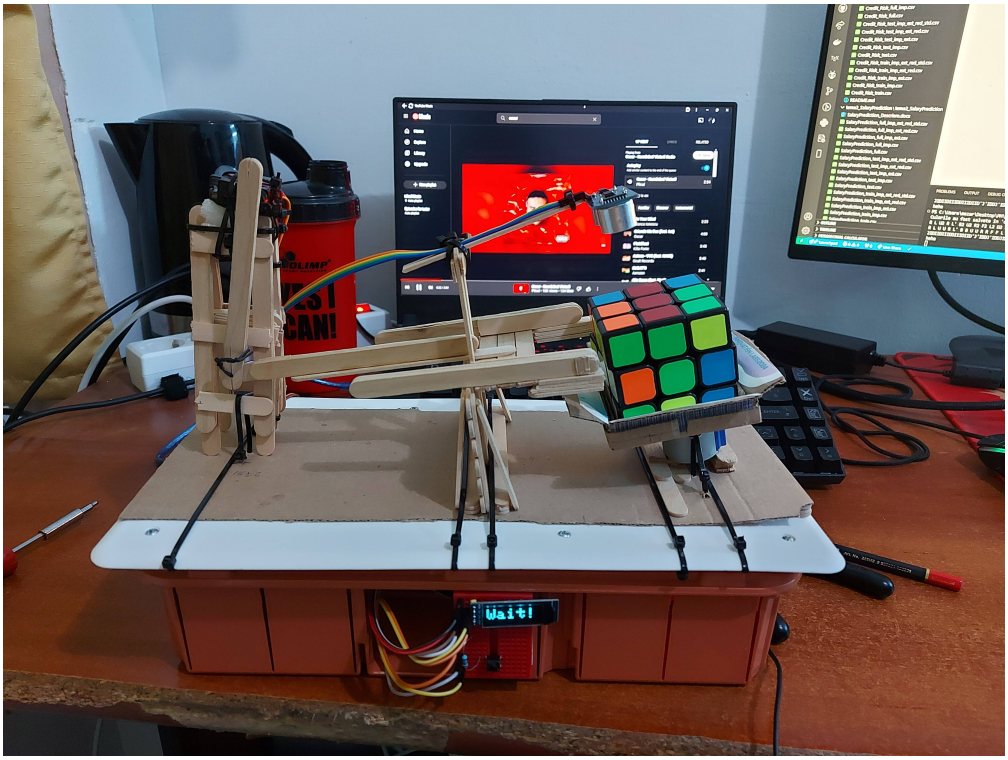
Rezultate Obţinute

Interfata grafica




Solution!
GND VCC SCL SDA





Demo

Exemplu de rulare

Concluzii

E greu sa faci un robot care rezolva un cub Rubik fara o imprimanta 3D

Download

Codul sursa al proiectului

Jurnal

5 Mai Activitate: Inițierea Proiectului CubeMaster Detalii: Am stabilit obiectivele proiectului și am început planificarea inițială a componentelor necesare. Am realizat cercetări preliminare privind algoritmii de rezolvare a Cubului Rubik și mecanismele automate de manipulare a cubului.

7 Mai Activitate: Construcția bazei pentru CubeMaster Detalii: Am început să construiesc baza fizică a CubeMaster. Am creat un cadru robust pentru a susține cubul Rubik și mecanismele de manipulare.

Am discutat designul general și am decis să folosesc un servomotor și un motor pas cu pas pentru manipularea cubului.

8 Mai Activitate: Preprocesare cu OpenCV Detalii: Am început dezvoltarea componentei de preprocesare pentru captarea fețelor cubului Rubik utilizând biblioteca OpenCV. Am configurat camera și am scris codul inițial pentru capturarea și identificarea culorilor fetelor cubului. Din păcate, am întâmpinat probleme cu calitatea camerei, culorile fiind dificil de distins. Galbenul apărea alb în imagini, ceea ce făcea recunoașterea culorilor ineficientă. Am decis să schimb abordarea și să mă concentrez pe alte părți ale proiectului.

19 Mai Activitate: Dezvoltarea interfeței grafice Detalii: Am creat interfața grafică utilizând tkinter pentru inputul manual al configurației fetelor cubului. Am adăugat butoane pentru a facilita introducerea culorilor și pentru a trimite datele către algoritmul de rezolvare. Am integrat biblioteca kociemba pentru generarea soluției optime de rezolvare a cubului Rubik. Am implementat funcții pentru preprocesarea stringurilor și conversia acestora în mutări specifice (D, I, J) pentru mecanismul CubeMaster.

20 Mai Activitate: Recepcionarea componentelor hardware și asamblare Detalii: Am primit servomotorul și alte componente hardware necesare. Am început asamblarea brațului robotic și a mecanismului de rezolvare. Am scris codul Arduino pentru controlul servomotorului și motorului pas cu pas. Am creat scheme electrice detaliate.

27 Mai Activitate: Testarea finală și optimizarea sistemului Detalii: Am realizat testele finale pentru sistemul CubeMaster. Am testat rotirea cubului și am rafinat mișcările pentru a asigura acuratețea și am închis toate componentele într-o cutie de siguranțe pentru a proteja și organiza hardware-ul. Am ajustat componentele și codul pentru a îmbunătăți funcționarea generală. De asemenea, am testat mecanismul de rezolvare pentru a ne asigura că toate mișcările sunt executate corect și eficient.

Bibliografie/Resurse

Cube Images
God's Number
Sursa de idee 1
Sursa de idee 2
Rezolvarea unui cub Rubik
Sursa de idee 3
Procesarea imaginilor pentru camera pe care nu am mai pus-o
Intelegere kociemba
Laboratoare + Cursuri

Export to PDF