Rubik's Cube Solver - Onciulescu Dragomir Anne Marie

Introducere

Acest proiect are ca scop dezvoltarea unui dispozitiv capabil sa rezolve un cub Rubik 3x3x3, pornind de la orice configuratie initiala valida.

Un solver de cub rubik foloseste in mod normal 6 steppere (cele din competitiile de speed solving robots), implementarea mea incearca o varianta diferita, folosind numai 3 steppere. (totusi nu pot spune ca as fi un pionier al acestei abordari)

Descriere generală

Miscarile posibile de actionare ale unui cub Rubik sunt urmatoarele: L (left clockwise), L' (left counterclockwise), R (right CW), R' (right CCW), F (front CW), F' (front CCW), B (back CW), B' (back CCW), U (up CW), U' (up CCW), D (down CW), D' (down CCW). Avand numai 3 steppere (left, right si back), doar 6 dintre aceste mutari sunt posibile as-is.

Pentru a putea executa celelalte miscari, se introduce un nou tip de miscare, rotateBackward si rotateForward. Acest tip de mutare este realizata prin actionarea simultana a stepperelor laterale. In urma acesteia, cubul va fi rotit cu totul in jurul axei formate de cele doua steppere diametral opuse. (de exemplu, fata Up va ajunge sa fie pozitionata unde se afla anterior fata Back, prin aplicarea unui rotateBackward sau in urma a 3 miscari rotateForward)

Procesul incepe prin introducerea intr-o interfacta grafica (pe laptop) a permutarii initiale a cubului. Datele sunt trimise catre un API ce intoarce, in urma aplicarii algoritmului Kociemba, o succesiune de pasi pentru rezolvare. Pasi sunt apoi trimisi catre Arduino pe seriala, unde vor fi transformati in miscari de actionare a stepperelor. Motorul TT este folosit pentru a stabiliza slice ul central al cubului in momentul rotirii individuale a fetelor laterale (altfel se agata).

Hardware Design

- Arduino Uno R3
- 3 x Nema-17 Stepper Motor (17HS4401)
- 3 x DRV8825 (Stepper Motor Driver)

- CNC Shield V3
- TT Gearbox Motor
- L298N H-Bridge
- Sursa de alimentare 12V 5A (pentru Drivere, prin CNC Shield)
- Sursa de alimentare 12V 2A (pentru H-Bridge)
- 2 x Conector Cub-Shaft Motor (3D printed)
- Conector Gheara-Shaft Motor (3D printed)
- ~~Custom PCB~~ (nu am mai avut timp sa il trimit la fabricat, a fost inlocuit de CNC shield)

Software Design

Functii

Functie Generica de Rotatie

Roteste un motor 90 de grade.

```
void rotateMotor(int stepPin, int dirPin, bool clockwise) {
  digitalWrite(dirPin, clockwise ? HIGH : LOW);
  for (int i = 0; i < STEPS_PER_90; i++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(STEP_DELAY_US);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(STEP_DELAY_US);
  }
}
```

Functii de rotatie

```
void rotateXCW()  { rotateMotor(X_STEP, X_DIR, true);  }
void rotateXCCW() { rotateMotor(X_STEP, X_DIR, false); }

void rotateYCW()  { rotateMotor(Y_STEP, Y_DIR, true);  }
void rotateYCCW() { rotateMotor(Y_STEP, Y_DIR, false); }

void rotateZCW()  { rotateMotor(Z_STEP, Z_DIR, true);  }
void rotateZCCW() { rotateMotor(Z_STEP, Z_DIR, false); }
```

Functie Generica de Rotatie Duala

Roteste intreg cubul 90 de grade pe axa L-R.

```
void DualRotateMotors(int stepPin1, int dirPin1, int stepPin2, int dirPin2,
bool direction) {
  digitalWrite(dirPin1, direction ? HIGH : LOW);
  digitalWrite(dirPin2, direction ? LOW : HIGH);
  for (int i = 0; i < STEPS_PER_90; i++) {
    digitalWrite(stepPin1, HIGH);
    digitalWrite(stepPin2, HIGH);
    delayMicroseconds(STEP_DELAY_US);
    digitalWrite(stepPin1, LOW);
    digitalWrite(stepPin2, LOW);
    delayMicroseconds(STEP_DELAY_US);
  }
}
```

Functii Rotatie Duala

```
void rotateForward()  { DualRotateMotors(X_STEP, X_DIR, Y_STEP, Y_DIR, true);
}
void rotateBackward() { DualRotateMotors(X_STEP, X_DIR, Y_STEP, Y_DIR, false);
}
```

Functie Translatare Mutari

Traduce miscarile cubului in rotatii ale motoarelor.

' – miscare CCW

2 – rotatie 180 de grade

```
void executeMove(String move) {
  bool twice = move.endsWith("2");
  bool prime = move.endsWith("'");

  char face = move.charAt(0);
  int turns = twice ? 2 : 1;

  for (int i = 0; i < turns; i++) {
    switch (face) {
      case 'R':
        if (prime) rotateXCCW();
        else rotateXCW();
        break;
```

```
      case 'L':
        if (prime) rotateYCCW();
        else rotateYCW();
        break;

      case 'B':
        if (prime) rotateZCCW();
        else rotateZCW();
        break;

      case 'F':
        rotateBackward();
        rotateBackward();
        if (prime) rotateZCCW();
        else rotateZCW();
        rotateForward();
        rotateForward();
        break;

      case 'U':
        rotateBackward();
        if (prime) rotateZCCW();
        else rotateZCW();
        rotateForward();
        break;

      case 'D':
        rotateForward();
        if (prime) rotateZCCW();
        else rotateZCW();
        rotateBackward();
        break;
    }
  }
}
```

Bucla Citire Mutari de pe Seriala

```
String incoming = "";

void loop() {
  while (Serial.available()) {
    char c = Serial.read();

    if (c == '\n' || c == '\r') {
      incoming.trim();
      if (incoming.length() > 0) {
```

```
        Serial.print("Received move: ");
        Serial.println(incoming);

        Serial.print("Executing: ");
        Serial.println(incoming);
        executeMove(incoming);

        Serial.println("OK");
      }
      incoming = "";
    } else {
      incoming += c;
    }
  }
}
```

sa revin asap cu restul detaliilor :))

asta e interfata grafica din python care rezolva cubul si dupa trimite pe seriala comenzile ptr motoare. solutiile de cub vin normal in forma unei liste ordonate de miscari din setul: U (up), D(down), L(left), R(right), F(front), B(back), U'(up reverse), U2(up de 2 ori) etc.

inainte sa fie trimsie catre arduino, respectiv catre motoare, trebuie aplicate niste transformari pentru a ramane in spatiul a 3 motoare ce pot fi controlate doar din dir si step. astfel o miscare de tip R ar fi straight forward, dar una precum U' ar insemna intai rotirea intregului cub cu fata de sus catre spate, prin actionarea celor doua motoare laterale simultan, urmat de o miscare de rotire a motorului din spate.

in imagine se poate vedea in dreapta interfata grafica in care se vaa introduce configuratia initiala a cubului. jos in terminal se vede solutia intoarsa dupa solve and send. in stanga este o solutie de pe un site de solver cub de pe net. se poate observa ca e acelasi raspuns.

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.