Cartografiere 3D a unui teren

Introducere

Proiectul propune realizarea unui sistem automatizat de cartografiere a suprafețelor, folosind un microprocesor Arduino, un senzor de distanță și un mecanism de scanare pe două axe. Măsurătorile realizate sunt afișate real-time prin intermediul unui ecran LCD, iar la final o aplicație specializată realizează un heat-map și o replică 3D a terenului scanat.

Scopul principal este de a construi un instrument portabil și intuitiv pentru analiza formelor de relief la scară mică, combinând vizualizarea imediată cu interpretarea detaliată pe calculator.

Ideea este inspirată dintr-o demonstrație văzută pe Internet, ce implica o hartă din nisip cu diferite forme de relief și un proiector care colora nisipul în diferite culori pentru a distinge între înălțimile terenului. Această aplicație era folosită pentru a învăța copiii să citească hărțile bazate pe înălțimi și adâncimi prin vizualizarea schimbărilor în timp real.

Consider că un astfel de proiect ar fi util atât pentru scopul prezentat mai sus, de prezentare interactivă a creării unei hărți, dar și pentru observarea imperfecțiunilor sau a diferențelor de nivel de pe o suprafață de dimensiuni reduse.

Descriere generală

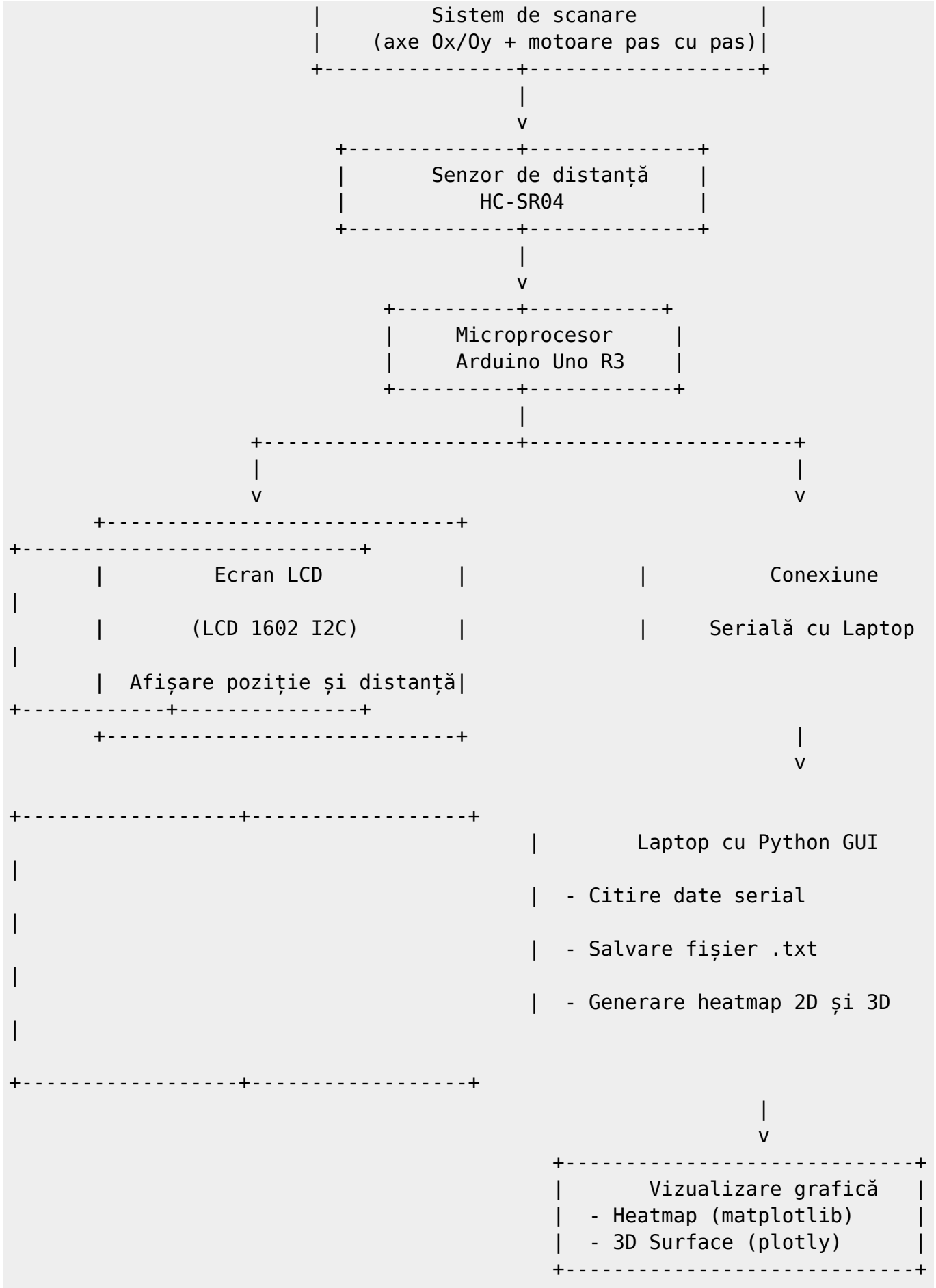
Componente hardware:

- Un microprocesor – pentru coordonarea tuturor operațiilor
- Un senzor de distanță – pentru măsurarea variațiilor de înălțime
- Un sistem de scanare pe două axe – bazat pe două motoare
- Un ecran LCD 1602 I2C – pentru reprezentarea real-time a datelor măsurate
- Un laptop– pentru prelucrarea extinsă a datelor și afișarea detaliată a heatmap-ului

Interacțiune hardware - software

- Scanare: Senzorul de distanță, montat pe un mecanism de scanare cu două axe, măsoară distanța față de suprafață în puncte discrete organizate sub forma unei matrice.
- Procesare locală: Aceste măsurători sunt transmise pe laptop, care realizează o serie de calcule pentru a transforma semnalul primit de la senzor în distanță de la suprafață până la senzor.
- Transmitere date: Valoarea fiecărei măsurători este afișată pe ecranul LCD, dar și în Serial Monitor, de unde un script de Python preia datele pentru a crea un heatmap și un model 3D
- Vizualizare digitală: Pe laptop, un software dedicat procesează datele primite și construiește un heat-map complet, dar și o replică 3D, oferind o reprezentare vizuală detaliată a terenului analizat.

```
                                +-----------------------------------+
```


Sistem de scanare
(axe Ox/Oy + motoare pas cu pas)
Senzor de distanță
HC-SR04
Microprocesor
Arduino Uno R3
Ecran LCD
(LCD 1602 I2C)
Afișare poziție și distanță
Conexiune
Serială cu Laptop
Laptop cu Python GUI
- Citire date serial
- Salvare fișier .txt
- Generare heatmap 2D și 3D
Vizualizare grafică
- Heatmap (matplotlib)
- 3D Surface (plotly)

Hardware Design

Listă de componente:

- 2 x motoare pas cu pas 28BYJ-48 cu drivere ULN2003
- 1 x senzor de distanță ultrasonic HC-SR04
- 1 x ecran LCD 1602 cu interfață I2C
- 1 x placă de dezvoltare compatibilă Arduino Uno R3
- 1 x breadboard
- 1 x laptop (pentru colectarea datelor și procesare)
- 1 x sistem de axe imprimat 3D
- Elemente de susținere realizate din carton (pentru structură și fixare)
- Fire pentru conectarea pinilor

Descrierea sistemului mecanic

Sistemul este proiectat pentru a realiza o scanare pe suprafața unei zone plane, utilizând un senzor de distanță ultrasonic montat pe o platformă mobilă în două direcții (Ox și Oy). Mișcarea bidimensională este asigurată de două motoare pas cu pas 28BYJ-48, fiecare fiind responsabil pentru deplasarea senzorului pe una dintre axe.

Mecanismul de transmisie pentru fiecare axă este realizat din roți dințate (gear train), integrate într-un cadru imprimat 3D. Axele au o lungime de aproximativ 30 cm. Axa Oy este fixată pe un suport din carton, astfel încât capetele axei rămân staționare, iar motorul de pe această axă deplasează întregul ansamblu mobil (slider) de-a lungul direcției verticale (în planul scanării).

Pe acest slider este montată și axa Ox, permițând astfel deplasarea senzorului și în direcția orizontală. Ansamblul funcționează ca un sistem de coordonate cartezian, în care senzorul este poziționat succesiv deasupra fiecărui punct dintr-o matrice 2D, în scopul măsurării distanței față de suprafață.

Modul de funcționare

Senzorul parcurge o matrice de 20×20 puncte de măsurare, urmând o traiectorie „linie cu linie”, similară cu următoarea ordine:

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 16

La fiecare punct, senzorul măsoară distanța față de suprafață, iar valorile sunt afișate în timp real pe un ecran LCD și transmise prin portul serial către un computer, unde pot fi salvate sub formă de fișier și ulterior prelucrate (de exemplu, pentru generarea unei hărți de înălțime).

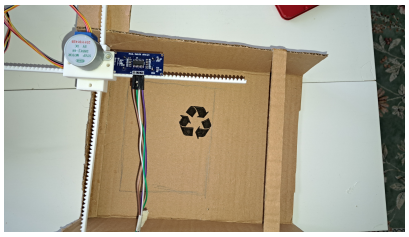
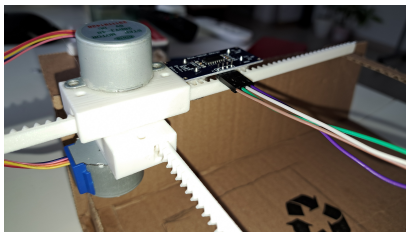
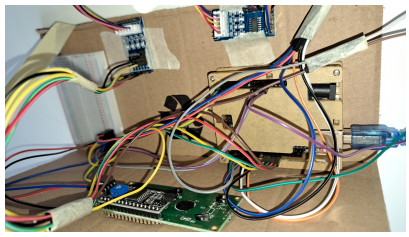
Estetică și stabilitatea mecanică

Un aspect important întâmpinat în timpul realizării sistemului a fost dezechilibrul axei Ox la extindere maximă, cauzat de greutatea motoarelor. Inițial, masa motoarelor afecta distribuția forței, determinând înclinarea platformei mobile. Problema a fost rezolvată prin reconfigurarea poziționării:

- Axa Oy a fost montată cu motorul orientat în jos, fixată într-un suport rigid de carton.
- Axa Ox a fost montată cu motorul în partea superioară a structurii, astfel încât greutatea acestuia să contribuie la stabilitate și să prevină balansarea nedorită.

Pentru a menține axa Ox cât mai paralelă cu suprafața de scanat, a fost adăugat un element de sprijin din carton, care funcționează ca o bară de susținere orizontală.

Componentele electronice, precum placa Arduino și breadboard-ul, au fost ascunse în spatele unui panou de carton , pe care este montat și afișajul LCD. Acest design asigură un aspect ordonat, oferind totodată protecție componentelor. Placa de dezvoltare este montată într-o carcasă de plexiglas pentru siguranță și vizibilitate parțială.

Software Design

Software-ul proiectului este împărțit în două componente:

- Codul Arduino – controlează mișcarea fizică a senzorului și colectează datele de la senzorul de distanță, pe care le transmite către un PC și le afișează pe un ecran LCD.
- Codul Python – procesează datele primite și generează vizualizări (heatmap 2D și reprezentare 3D a terenului).

Arduino

Biblioteci utilizate

- **Arduino.h** - inclusă automat și oferă suport pentru funcțiile setup() și loop().
- **LiquidCrystal_I2C.h** 1.1.4 - permite controlul unui ecran LCD 1602 cu interfață I2C, ceea ce reduce numărul de pini folosiți pe Arduino.
- **Stepper.h** 1.1.3 - biblioteca standard pentru controlul motoarelor pas cu pas în configurație 4-wire.
- **Wire.h** - gestionează protocolul I2C pentru comunicarea cu ecranul LCD. Este necesară implicit pentru LiquidCrystal_I2C.
- **EEPROM.h** - permite citirea și scrierea în memoria EEPROM a plăcii Arduino.

Variabile globale și inițializare

- LiquidCrystal_I2C lcd(0x27, 16, 2) - inițializează LCD-ul I2C pe adresa 0x27, cu 2 rânduri și 16 coloane.
- const int trigPin = 3, echoPin = 2 - pinii conectați la senzorul HC-SR04 pentru generarea și recepția semnalului ultrasonic.
- Stepper motorX(...), motorY(...) - motoarele pas cu pas pentru deplasarea pe axele X și Y.
- const int gridSize = 20 - dimensiunea matricei de scanare (20×20 puncte).
- stepPerCellX și stepPerCellY - numărul de pași necesari pentru ca senzorul să se deplaseze o poziție în grid.

- currentX, currentY - poziția curentă a senzorului în grid, salvată în EEPROM.

Funcția setup()

- Inițializează comunicarea serială, ecranul LCD, motoarele și pinii senzorului.
- Citește poziția salvată anterior din EEPROM pentru a putea relua o scanare întreruptă.
- Afișează pe ecran poziția de la care se va relua scanarea („Resume from:”).

Funcția loop()

Parcurgerea matricei:

- Se scanează linie cu linie.
- Dacă s-a întrerupt anterior, se reia exact de unde a rămas.

```
for (int y = currentY; y < gridSize; y++) { for (int x = (y == currentY ? currentX : 0); x < gridSize; x++) {
```

Măsurarea distanței:

- Se trimite un semnal ultrasonic și se măsoară timpul până la recepția ecoului.
- Se calculează distanța în centimetri.

```
do { digitalWrite(trigPin, LOW); delayMicroseconds(2); digitalWrite(trigPin, HIGH); delayMicroseconds(10); digitalWrite(trigPin, LOW); duration = pulseIn(echoPin, HIGH); distance = duration * 0.034 / 2;
```

- Distanța este recalculată relativ la înălțimea de montare (8 cm), pentru a determina „relieful”.

```
real_distance = 8 - distance;
```

- Se aplică un prag de schimbare: dacă noua valoare diferă cu ⇐ 1 cm de cea anterioară, se consideră aceeași.

```
if (lastDistance != 0) { if (real_distance - lastDistance < 2 && real_distance - lastDistance > -2) real_distance = lastDistance; }
```

Afișarea și trimiterea de date:

- Se afișează coordonatele și distanța curentă.

```
lcd.clear(); lcd.setCursor(0, 0); lcd.print(“X:”); lcd.print(x); lcd.print(” Y:”); lcd.print(y); lcd.setCursor(0, 1); lcd.print(“Dist: ”); lcd.print(real_distance); lcd.print(” cm”);
```

- Se trimit datele în format CSV-like către un fișier de pe laptop prin serial (pentru prelucrare ulterioară în Python).

```
Serial.print(x); Serial.print(” ”); Serial.print(y); Serial.print(” ”); Serial.println(real_distance);
```

Salvarea progresului în EEPROM:

- După fiecare punct scanat, se salvează poziția următoare în EEPROM, pentru a permite reluarea în

caz de resetare.

```
EEPROM.put(0, x + 1); EEPROM.put(sizeof(int), y);
```

Mișcarea senzorului:

- **motorX.step(-stepPerCellX)** – deplasare spre stânga după fiecare măsurare.

La final de rând:

- **motorX.step(stepPerCellX * gridSize)** – revine la începutul rândului.
- **motorY.step(stepPerCellY)** – se deplasează o poziție pe axa Y.

Finalizarea scanării

După finalizarea completă a gridului, se șterg datele salvate în EEPROM și se mută senzorul înapoi în poziția de start (0, 0).

Python

Configurarea serială

- Se definesc portul serial și rata de transfer pentru comunicarea cu Arduino. GRID_SIZE trebuie să corespundă cu dimensiunea definită în codul Arduino.

```
SERIAL_PORT = 'COM5' BAUD_RATE = 9600 GRID_SIZE = 20
```

Inițializare matrice și conexiune serială

- Se creează o matrice de 20×20 care va fi completată cu valorile măsurate.
- Se inițializează conexiunea serială.
- Pauza de 2 secunde permite Arduino-ului să repornească și să fie gata de transmisie.

Colectarea datelor de la Arduino

- Se citește fiecare linie trimisă prin Serial.print() de Arduino.
- Se parsează poziția X, Y și distanța măsurată.
- Datele sunt stocate în matrice, inversând pozițiile pentru ca linia să corespundă axei Y și coloana axei X.
- Când poziția (19, 19) este atinsă, se consideră că matricea este completă.

Salvarea datelor într-un fișier

- Datele sunt salvate într-un fișier .txt pentru utilizare ulterioară.

Vizualizare – Heatmap

- Se convertesc datele în format *NumPy* pentru procesare rapidă.
- Se „aplatizează” datele pentru a putea fi procesate de funcția *gaussian_kde()* din *scipy*.
- Se folosește o funcție de densitate gaussiană pentru a obține o suprafață netedă bazată pe înălțimi: *weights=z* înseamnă că punctele cu înălțime mai mare influențează mai mult forma finală.
- Se creează două heatmap-uri: unul pentru a avea legendă (*cbar*) corectă, altul pentru reprezentarea grafică propriu-zisă.
- Se adaugă contururi (*contour*) pentru claritate.

- Se inversează axa Y pentru ca (0,0) să fie în colțul stânga-sus.

Vizualizare - Reprezentare 3D interactivă

- Se folosește *plotly.graph_objects* pentru a genera o suprafață 3D.
- *colorscale*='*Plasma*' menține coerența culorilor cu heatmap-ul 2D.
- Setările includ titlurile axelor, scalarea și aspectul 3D.
- Utilizatorul poate roti și mări/mișca suprafața cu mouse-ul.

În final, după scanarea terenului, programul afișează automat cele două grafice ce prezintă schimbările de formă ale terenului țintă.

Rezultate Obţinute

În urma implementării și testării sistemului de scanare, rezultatele au fost în concordanță cu obiectivele stabilite. Dispozitivul a fost capabil să scaneze o suprafață plană discretizată într-o matrice de 20×20 de puncte, colectând în total 400 de valori de distanță cu ajutorul senzorului HC-SR04. Aceste valori au fost prelucrate și afișate atât local, pe un ecran LCD, cât și transmise prin port serial către un laptop pentru analiză și vizualizare.

Din datele colectate, s-a generat o hartă 2D care reflectă variațiile de înălțime relativă pe suprafața scanată, precum și o reprezentare tridimensională interactivă a terenului. Aceste vizualizări au permis observarea rapidă a eventualelor denivelări, dovedind astfel utilitatea sistemului în reprezentarea spațială a unei suprafețe scanate.

Pe parcursul testării, sistemul a demonstrat o repetabilitate satisfăcătoare a măsurătorilor, iar algoritmul de reluare automată a scanării (prin EEPROM) a funcționat conform așteptărilor, permițând reluarea scanării de la ultimul punct în cazul unei întreruperi neprevăzute. Astfel, s-a confirmat că proiectul nu doar captează date relevante, ci și le structurează eficient pentru prelucrare ulterioară.

Concluzii

Proiectul se remarcă printr-o integrare eficientă a componentelor hardware și software, reușind să automatizeze complet procesul de scanare a unei suprafețe printr-un sistem de axe coordonat de motoare pas cu pas. Utilizarea senzorului de distanță HC-SR04 permite măsurători precise la fiecare punct al grilei, iar afișajul LCD oferă feedback vizual în timp real asupra poziției și distanței măsurate. Persistența datelor este asigurată prin salvarea poziției curente în EEPROM, astfel încât întreruperile neașteptate ale alimentării nu afectează progresul scanării. Partea software dezvoltată în Python aduce o valoare adăugată semnificativă prin vizualizările generate. Datele colectate sunt convertite într-un heatmap 2D și o reprezentare 3D interactivă a reliefului, ceea ce face analiza datelor intuitivă și estetică.

Limitări ale proiectului

Cu toate aceste realizări, proiectul întâmpină și unele limitări. Senzorul HC-SR04, deși accesibil și ușor de folosit, are o precizie relativă scăzută în medii instabile sau la suprafețe neregulate, ceea ce poate

duce la fluctuații nedorite în măsurători. Algoritmul de filtrare implementat este simplu și nu elimină complet erorile, fiind loc pentru îmbunătățiri prin metode mai avansate. Viteza de scanare este limitată de caracteristicile hardware și de dimensiunea gridului, ceea ce face dificilă extinderea proiectului pe suprafețe mai mari fără optimizări suplimentare.

Download

Link cod - Github

Pentru ca scriptul din Python să poate prelua datele din Serial Monitor trebuie ca, imediat după încărcarea codului pe plăcuță, să fie pornit si fișierul cu cod Python.

Jurnal

În această secțiune voi enumera problemele întâmpinate pe parcursul proiectului, dar și soluțiile găsite în cazul în care problema a fost rezolvată:

- **plăcuța nu mai accepta upload de cod din Platformio** - la un moment dat, nu mai puteam da upload la codul din Platformio pe plăcuță, chiar dacă operațiunea de build era successful; problema este una recurentă în special pentru plăcuțele compatibile cu Arduino, dar neoriginale, iar soluția am găsit-o pe un forum dedicat Arduino, aceasta fiind apăsarea butonului de reset de pe plăcuță cât timp se execută primii pași din procesul de upload
- **matricea de LED-uri nu se aprindea** - înainte de a apela la un ecran LCD penrtu afișare, am avut în plan să afișez heatmap-ul atât pe laptop, cât și pe o matrice de LED-uri conectată la plăcuță, dar din cauza faptului că această matrice, pentru a funcționa, avea nevoie de o bibliotecă din Python ce nu putea fi instalată (probabil pentru că nu era updatată de câțiva ani), nu am reușit să o fac să funcționeze
- **amplasarea motoarelor pe axele printate 3D** - atât problema, cât și soluția au fost detaliate în partea de Hardware Design
- **contrastul ecranului LCD** - ecranul LCD pe care l-am achiziționat are un șurub pe spate care reglează contrastul ecranului, iar acesta a fost livrat cu contrastul setat pe 0 (ecranul nu afișa nimic vizibil); soluția pentru reglarea ecranului și faptul că acesta, din fabrică, este setat pe are contratul setat pe 0 le-am aflat căutând datasheet-ul modelului de ecran

Bibliografie/Resurse

- Documentație motor pas cu pas și modul
- Documentație senzor de distanță
- Documentație ecran LCD
- Comunicare Arduino - Python

Export to PDF