Light Follower Robot - Bloțiu Mihnea-Andrei - 333CA

Demo Proiect

Introducere

Ideea principală

Proiectul constă în realizarea unui robot ce are scopul de a urmări o sursă de lumină ținând cont de intensitatea, poziția și direcția de propagare a acesteia.

Motivație

Dorința pentru implementarea acestui proiect a apărut acum mulți ani când am participat la un curs de testare/vizualizare a unor roboți realizați de către persoane pasionate de acest domeniu. Mereu am fost interesat și de modalitatea prin care aceștia au realizat roboții. Curiozitatea mea a crescut, iar aceasta fost o oportunitate perfectă pentru a vedea și partea implementării hardware și respectiv software aflată înaintea celei sesizate de mine în trecut.

Descriere generală

Explicarea detaliată a proiectului

- Proiectul presupune realizarea unui robot ce conține **trei module LDR (Light Dependent Resistor)** ce vor capta intensitatea unei surse de lumină. Acestea vor transmite datele captate către plăcuța Arduino, ce va inița motoarele ce la rândul lor vor deplasa robotul într-o anumită direcție.
- Deplasarea robotului se va realiza folosind un algoritm de tip **PID (Proportional, Integral and Derivative)** ce va ține cont de intensitatea luminoasă pe care o captează fiecare dintre cele trei

module LDR (ce vor fi plasate în poziții diferite). Astfel, robotul va putea păstra o singură direcție sau va putea vira în funcție de orientarea sursei de lumină. De exemplu, pentru ca robotul să meargă drept, există două situații. Fie doar modulul LDR din mijloc va detecta lumina, fie toate cele 3 module vor detecta lumina.
- Motoarele de pe aceeași parte a robotului vor fi **legate în serie** deoarece deplasarea din punct de vedere fizic nu ar avea sens altfel.
- Vor exista două surse diferite de alimentare pentru că motoarele consumă suficient de mult cât să nu poată fi comandate dintr-o baterie de 9V ce ar alimenta atât plăcuța Arduino cât și L293D. Astfel se va folosi bateria de 9V doar pentru alimentarea plăcuței Arduino, iar pentru alimentarea driver-ului de motoare vom folosi 3 acumulatori Li-Ion de 3.7V fiecare.
- Din punct de vedere software se va ține cont și de **accelerația/viteza la un anumit moment de timp** a robotului și deci, nu vor exista doar două stări ale acestuia (de deplasare și de repaus), ci va accelera/decelera treptat din două puncte de vedere:
  1. Motoarele folosite **nu au posibilitatea de a frâna**, deci atât accelerarea cât și decelerarea se vor realiza treptat;
  2. În algoritmul PID, **ținem cont de cât de mult timp s-a realizat deplasarea robotului într-o anumită direcție** în așa fel încât la schimbarea bruscă a direcției, robotul va continua să meargă pentru puțin timp în direcția inițială (din ce în ce mai puțin) până când într-adevăr se va adapta la noua direcție de propagare a luminii.
- În final, proiectul presupune și **utilizarea unui LCD** pentru afișarea bateriei rămase la un anumit moment de timp.

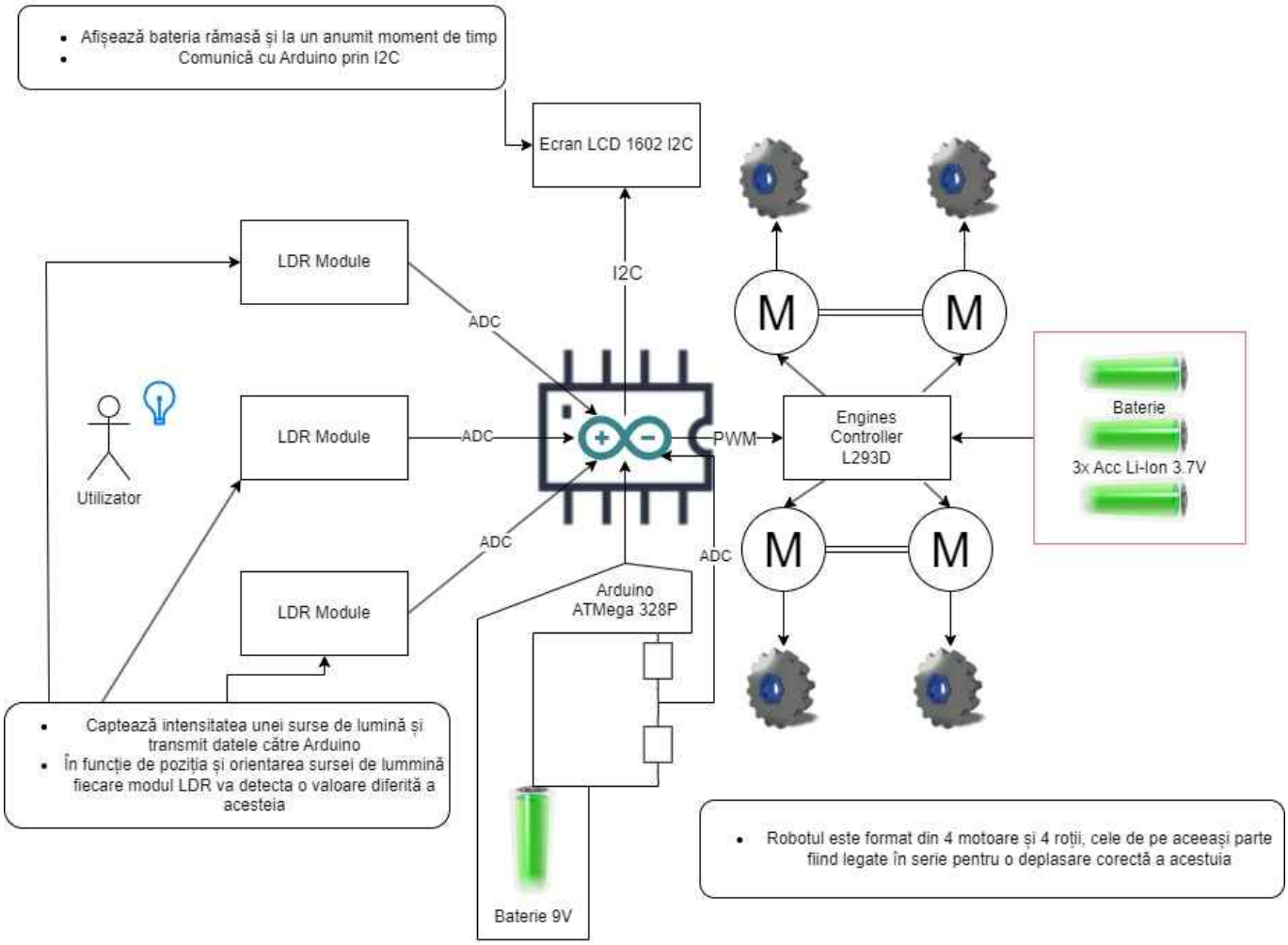
Laboratoare utilizate în dezvoltarea proiectului

În realizarea proiectului au fost utilizate noțiuni din **cel puțin trei laboratoare** prezentate la materia de **Proiectarea cu Microprocesoare** precum:

- Laboratorul 3 - PWM pentru transmiterea semnalelor către controller-ul de motoare
- Laboratorul 4 - ADC pentru interpretarea semnalelor de la modulele LDR si a voltajului de la baterie
- Laboratorul 6 - I2C pentru transmiterea datelor către LCD

Schema bloc a proiectului

O privire de ansamblu asupra proiectului menționat anterior poate fi compactată în următoarea imagine ce vorbește de la sine:



Hardware Design

Piese utilizate pentru realizarea proiectului

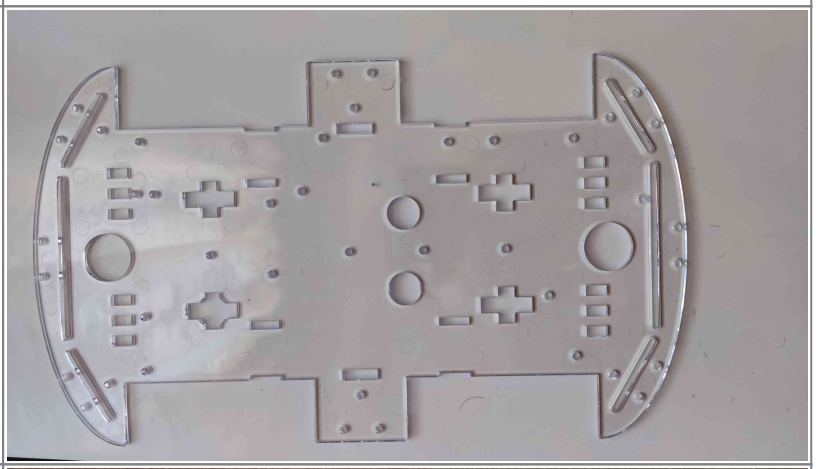
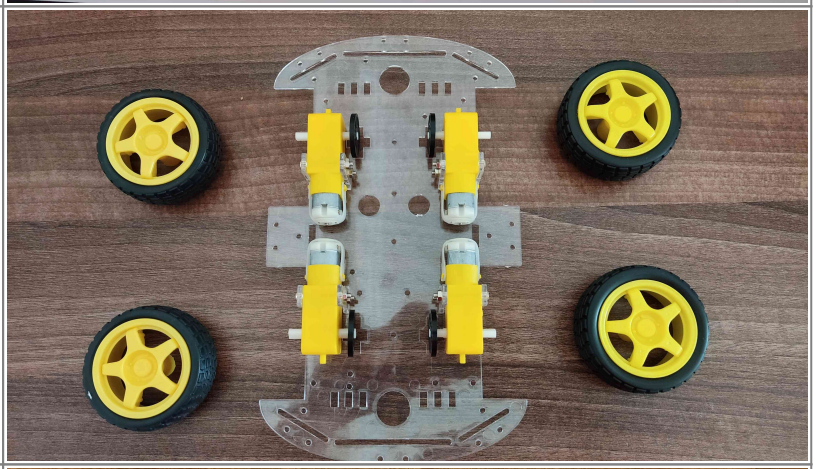
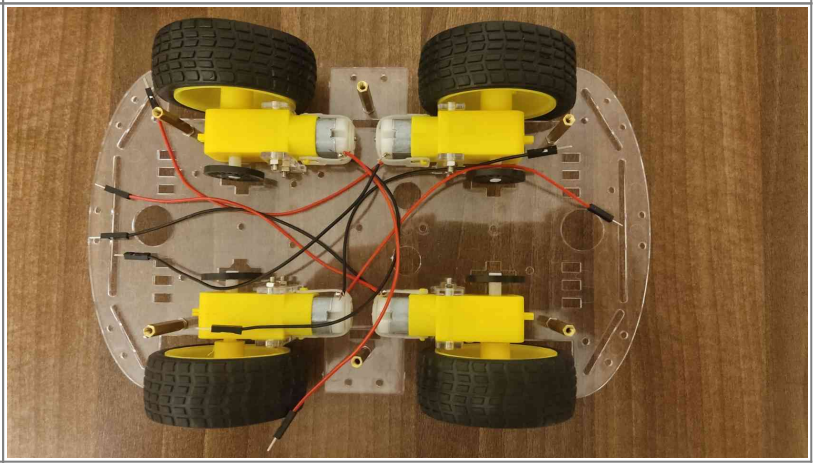
Următoarele piese au fost utilizate pentru realizarea integrală a proiectului:

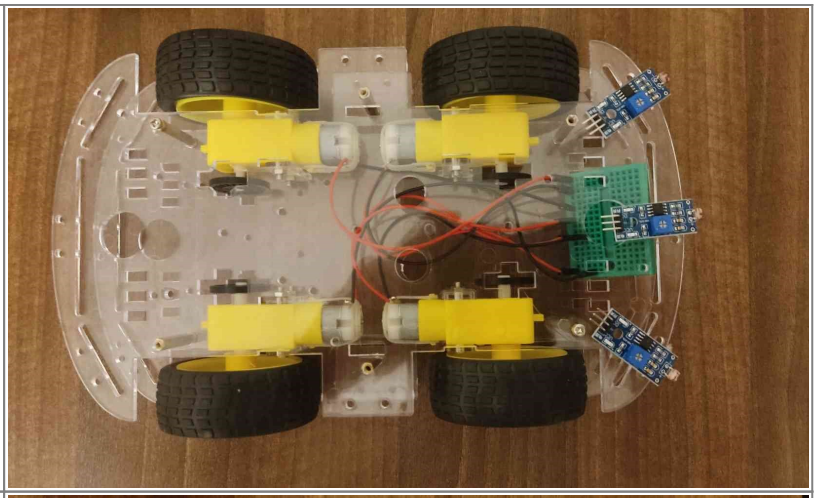
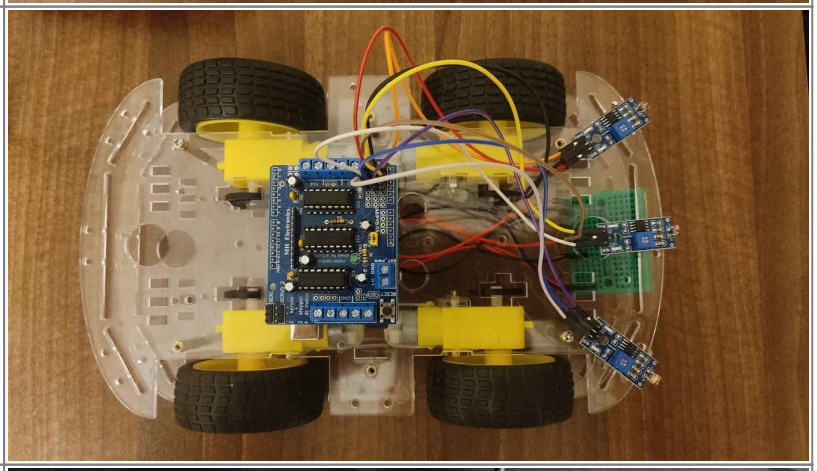
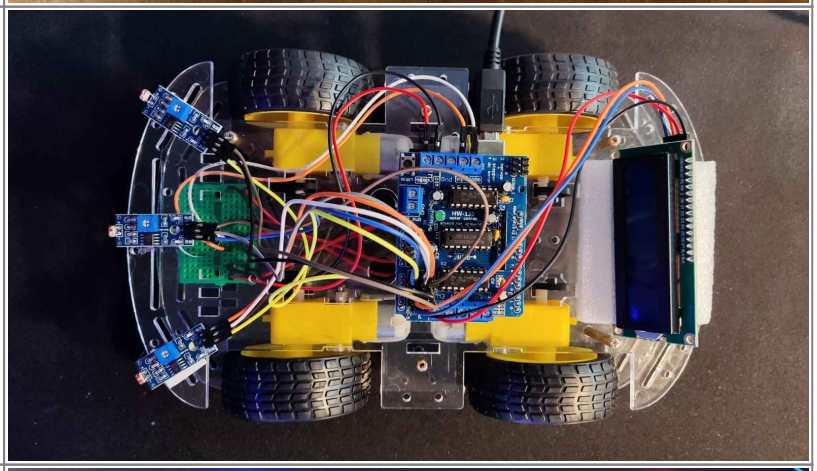
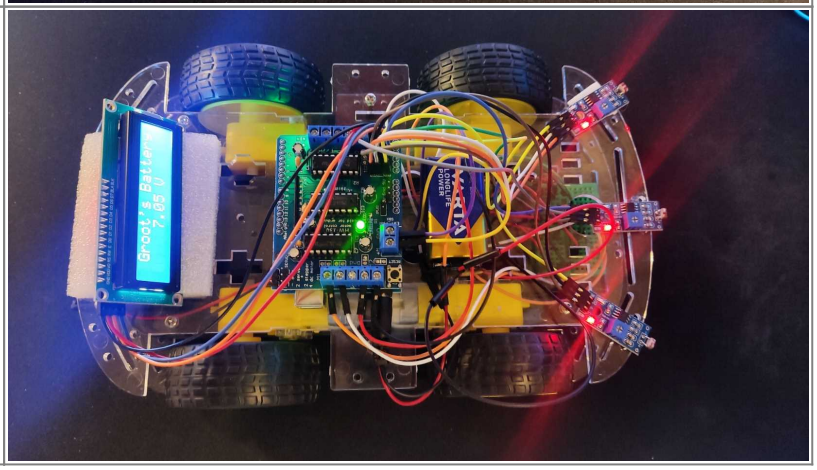
- 1x - Arduino UNO - ATMega328P
- 2x - Șasiu robot
- 4x - Roată robot
- 4x - Motor cu reductor - 3-6V
- 1x - Placă control motoare - L293D
- 1x - Breadboard mini - 170 puncte
- 1x - Suport baterii 3xAA
- 1x - Ecran LCD 1602 I2C
- 3x - Modul LDR
- 3x - Acumulator 3.7V
- 1x - Baterie 9V
- 1x - Pistol de lipit cu cositor

- 1x - Pistol de lipit cu silicon
- Fire de legătură

Pași dezvoltare Hardware

Rezultatele obținute pe parcursul dezvoltării hardware a proiectului vor fi prezentate succesiv în următorul tabel:

Etapă	Dovadă foto
Șasiu inițial	
Motoare montate	
Motoare + roți montate	

Montat Senzori LDR	
Montat Arduino + L293D	
Montat LCD	
Varianta Finala	

Software Design

Prezentarea uneltelor folosite

- Mediu de dezvoltare folosit: **Arduino IDE**
- Pentru realizarea proiectului au fost realizate atât **operații direct cu registre (pentru ADC)** cât și folosind următoarele biblioteci:
  1. **LiquidCrystal_I2C** - pentru interacțiunea cu ecranul LCD prin I2C
  2. **AFMotor** - pentru controlul celor 4 motoare prin semnale PWM

Prezentarea software-ului implementat

Proiectul curent este împărțit în două secțiuni principale ce vor fi descrise separat în cele ce urmează:

- Afișarea diferenței de potențial de la bornele bateriei de 9V pe LCD;
- Algoritmul PID pentru controlul robotului pe baza valorilor transmise de senzori;

Afișarea diferenței de potențial de la bornele bateriei de 9V pe LCD

Pentru realizarea acestui obiectiv s-au realizat următorii pași:

- Din punct de vedere hardware, am fost nevoit să realizez un **divizor rezistiv** cu două rezistențe de 10k asupra tensiunii de 9V în așa fel încât să mă asigur că pe pinul analogic al plăcuței Arduino nu o să ajungă niciodată o **tensiune mai mare de 5V** (maxim suportată).
- Apoi, această tensiune ce putea fi **maxim 4.5V** este **citită folosind ADC-ul** pus la dispoziție de Arduino o dată la 10 secunde. Un timp mai scurt **nu ar fi avut sens deoarece valoarea de la bornele bateriei nu se schimbă așa de des**.
- Această citire se face **configurând ADC-ul pe biți să citească semnalul de pe pinul A3 cu tensiunea de referință de 5V**, detaliile fiind exprimate în scheletul de cod pentru fiecare instrucțiune în parte.
- După citire, trebuie să facem conversia necesară în așa fel încât să aflăm **tensiunea reală de la bornele bateriei**. Cum valoarea din **ADC este egală cu tensiunea de intrare * 1023 / tensiunea de referință, putem afla tensiunea inițială ca fiind valoarea din ADC * 5 / 1023.**
- În acest moment, de aducem aminte de faptul că semnalul de intrare în pinul A3 nu este tensiunea de la bornele bateriei ci **este jumătate din aceasta** deoarece am realizat un divizor rezistiv. În acest sens, **pentru a afla rezultatul final, trebuie să înmulțim cu 2 valoarea obținută mai devreme**.
- În final, valoarea obținută mai devreme o să fie **afișată prin I2C** alături de numele robotului pe ecranul LCD-ului folosind biblioteca menționată mai devreme (**LiquidCrystal_I2C**). Această operațiune este realizată folosind metodele de **setCursor, și print** disponibile în bibliotecă.

Algoritmul PID pentru controlul robotului pe baza valorilor transmise de senzori

Pentru realizarea acestui obiectiv s-au realizat următorii pași:

- În primul rând **citim de la modulele LDR** o dată pe secundă (deoarece valorile transmise de senzori sunt mult mai volatile decât diferența de potențial de la bornele bateriei) valorile transmise de acestea pe pinii analogici A0, A1 și A2. Chiar dacă modulele LDR transmit pe ieșire semnale digitale, eu nu aveam la dispoziție decât pini analogici așa încât a trebuit folosit din nou ADC-ul pe biți pentru citirea semnalelor trimise de acestea. Astfel, **experimental am observat** că în cazul în care senzorii detectau lumină, aceștia raportau o valoare **în jur de 40**, iar în cazul în care nu detectau lumină **o valoare în jur de 1023**.
- Astfel, **am configurat pe rând ADC-ul pe biți** pentru a citi valorile de la cei 3 senzori și în funcție de valoarea transmisă de aceștia am **plasat într-un vector cu 3 poziții valoarea LOW sau HIGH**, unde prima poziție era atribuită senzorului din stânga, a doua celui din mijloc și a treia celui din dreapta.
- De menționat aici este faptul că pentru modificarea canalului pe care se primește semnalul analogic, noi schimbăm valorile biților de MUX din registrul ADMUX. Aici, ne bazăm pe faptul că **indiferent dacă se dorește sau nu o modificare a acestor biți, rezultatul nu va avea efect până când conversia anterioară nu a fost finalizată.**
- În acest moment, avem într-un vector, cele 3 valori de LOW sau HIGH pentru fiecare dintre cei 3 senzori, deci ne putem apuca să implementăm propriu zis algoritmul de PID.
- Pentru început, algoritmul PID se bazează pe **determinarea unei erori de poziționare a robotului față de direcția corectă de deplasare a acestuia**. În cazul nostru, considerăm direcția corectă de deplasare a acestuia ca fiind cea perfect dreaptă în care **toate cele 4 motoare au aceeași turație**. Această stare corectă de deplasare, poate fi atinsă în **două situații**: fie când toți cei 3 senzori detectează lumină, fie când doar senzorul din mijloc detectează lumină. Astfel, trebuie să ne gândim la **o formulă** prin care să calculăm starea corectă a robotului în așa fel încât **cele două situații menționate anterior să returneze aceeași valoare**.
- Astfel, **definim starea curentă a robotului** ca fiind (0 * valoareaSenzorului0 + 1000 * valoareaSenzorului1 + 2000 * valoareaSenzorului2) / (suma valorilor senzorilor) unde senzorul 0 este cel din stânga, senzorul 1 cel din mijloc, iar senzorul 2 cel din dreapta. Astfel, dacă toți 3 senzorii au valoarea HIGH (adică întâlnesc lumină), sau dacă doar senzorul din mijloc are valoarea HIGH, **atunci rezultatul stării curente va fi același adică 1000**.
- Calculând această valoare o dată la fiecare secundă, putem să calculăm **eroarea robotului (practic virajul robotului)** ca fiind valoarea pentru a se deplasa drept (1000) - valoarea curentă a vectorului.
- Pe baza acestei erori, calculăm cele 3 componente ale algoritmului PID (proportional, derivative and integral) și obținem **puterea pe care trebuie să o adăugăm/scădem de pe una dintre cele două grupări de motoare** ale robotului. De asemenea trebuie să avem grijă să nu depășim o viteză maximă a robotului.
- Cu valoarea finală a puterii ca fiind **suma celor 3 componente menționate anterior**, folosim biblioteca **AFMotor** și mai exact metodele de **run și setSpeed** pentru a controla viteza și sensul de rulare al motoarelor.

Rezultate Obţinute

Din punctul meu de vedere, următoarele au fost câteva rezultate personale obținute în urma finalizării proiectului

- Pentru mine, aceasta a fost **prima interacțiune de lungă durată pe care am avut-o cu Arduino**, deci a fost primul proiect realizat integral de către mine pe acest domeniu. În acest sens, eu mă declar mândru de robotul final. Am reușit să obțin un robot ce urmărește suficient de bine o sursă de lumină, să implementez un algoritm PID despre care am aflat de abia în acest semestru și nu în ultimul rând să îmi satisfac curiozitatea menționată în zona de introducere a acestei pagini
- Cel mai important rezultat pentru mine este acela că fiind primul proiect de acest tip cu care am interacționat, m-am lovit de nenumărate probleme, pe care am reușit să le duc într-o maniera **„good enough”** la final, dar într-un timp de aproximativ 2 luni de zile. Astfel, sunt convins că dacă pe viitor o să mai realizez un astfel de robot, timpul în care îl voi aduce la forma finală va fi unul **mult mai scăzut**

Concluzii - Dificultăți întâmpinate și îmbunătățiri posibile

Următoarea listă reprezintă câteva problemele cu care eu m-am întâlnit de-a lungul realizării proiectului și cum le-am soluționat. De asemenea, voi încerca să menționez și câteva îmbunătățiri pe care doresc să le aduc cât de curând:

- În primul rând, de departe cea mai dificilă parte cu care m-am lovit a fost cea legată de alimentarea proiectului. Încercând să mă documentez cât mai mult posibil, am văzut foarte multe proiecte similare care funcționau foarte bine cu o singură baterie de 9V. Am încercat și eu același lucru însă efectiv nu exista suficientă putere pentru ca aceasta să alimenteze întreg circuitul. După ce mi-am dat seama că nu există nicio șansă cu o singură baterie de 9V, am aflat că ar fi bine să am surse de alimentare separate: una pentru Arduino și alta pentru driver-ul de motoare L293D, ceea ce am și făcut și pare că proiectul rulează fără probleme. Totuși, am pierdut destul de mult timp cu asta și am ajuns la o variantă finală mai degrabă experimental. Pe viitor chiar mi-ar plăcea să am o modalitate mult mai formală pentru a calcula ce sursă de alimentare am nevoie pentru un anumit proiect.
- O altă problemă mai mică de care m-am lovit a fost faptul că inițial am crezut fără o cercetare prea atentă că modulele LDR pe care eu le-am comandat transmit pe ieșire un semnal analogic, ceea ce s-a dovedit a fi fals în momentul în care am vrut să afișez valoarea pe care o transmit acestea pe pinurile analogice. Nu a fost nicio problemă din punct de vedere al implementării finale a proiectului, dar în acest sens, pe viitor vreau să modific aceste module LDR cu altele ce transmit pe ieșire o valoare analogică pentru ca algoritmul PID implementat de mine să aibă un rezultat și mai precis.

Download

Codul sursă comentat în Arduino IDE, schema și pozele de mai sus pot fi găsite în următoarea arhivă: pm-blotiu-mihnea-andrei-333ca-2023.rar

Bibliografie/Resurse

Resurse documentație

Următoarele resurse au fost utilizate pentru scrierea documentației pe pagina curentă:

Documentație Wiki OCW
Tool pentru realizarea schemei bloc

Resurse hardware și/sau software

Următoarele resurse au fost utilizate pentru inspirație hardware/software:

Tutorial pentru măsurarea diferenței de potențial la bornele bateriei de 9V
Tutorial pentru realizarea unui algoritm PID pentru un robot maze
Tutorial pentru algoritmul PID în general
Tutorial pentru montarea pieselor robotului
Datasheet ATMega328P
Documentație bibliotecă LiquidCrystal_I2C
Documentație bibliotecă AFMotor

Export to PDF