Line Following Robot

Mureşan Alexandra

Takacs Răzvan

30239

Cuprins

1. Introducere

Acest proiect se concentrează pe proiectarea și implementarea unui robot de urmărire a liniei utilizând platforma Arduino Uno. Unitatea este echipată cu doi senzori de tip infraroșu (IR) pentru identificarea traseului și este pusă în mișcare de două motoare interconectate printr-o structură de tip punte H. Obiectivul principal este elaborarea și aplicarea unui algoritm avansat care să permită robotului să finalizeze un circuit printr-o navigare precisă de-a lungul unei linii prestabilite.

1.1 Motivarea alegerii

Alegerea acestui proiect este motivată de relevanța sa în contextul actual, întrucât numeroase tehnologii contemporane se bazează pe utilizarea senzorilor și pe procesarea diverselor informații colectate din mediul înconjurător. Prin intermediul acestui proiect, ne propunem să explorăm și să ne integrăm în domeniul dinamic al roboticii. Acest gen de proiect este frecvent întâlnit în sfera Arduino și a roboticii. Totuși, am urmărit să adoptăm o perspectivă ușor diferită, integrând detalii specifice în implementarea sa pentru a-i conferi un caracter distinctiv și inovator.

2. Soluția propusă

Soluția propusă de noi este relativ simplă și eficientă: pe cadru, robotul este echipat cu doi senzori infraroșii de urmărire a liniei, orientați către suprafața de deplasare. Funcția principală a acestor senzori este de a emite un semnal atunci când detectează linia.

2.1 Funcționalități

În funcție de datele primite, gestionăm patru scenarii diferite:

1. Când ambii senzori transmit semnalul 0, motoarele vor fi activate la aceeași viteză, permițând robotului să se deplaseze în linie dreaptă.
2. Dacă senzorul din stânga emite semnalul 1, iar cel din dreapta 0, motorul drept va fi alimentat cu o putere mai mare, în timp ce cel stâng va primi mai puțină energie, determinând astfel virajul spre stânga.
3. În mod similar, pentru virajul la dreapta, când senzorul din dreapta emite semnalul 1 și cel din stânga 0, motorul stâng va primi o putere mai mare, iar cel drept va fi redus.
4. În cazul în care ambii senzori transmit valoarea 1, ambele motoare se vor opri.

Această abordare simplifică procesul de decizie și control al robotului, făcându-l ideal pentru navigarea de bază în urmărirea liniei.

Un element distinctiv al proiectului nostru, care îl diferențiază de alte inițiative similare, este implementarea unui mecanism de pornire controlată prin intermediul unui buton situat pe cadru. Această caracteristică permite activarea robotului doar atunci când utilizatorul apasă explicit acest buton, oferind un control suplimentar și evitând pornirile accidentale sau nedorite. Această funcție adaugă un nivel suplimentar de interactivitate și siguranță, făcând ca robotul să fie mai adaptabil și utilizator-friendly în diferite contexte de utilizare.

În plus, robotul nostru este echipat cu o bandă Neopixel formată din 8 LED-uri, care oferă un feedback vizual atractiv și intuitiv, corespunzător diferitelor mișcări ale robotului. Acest sistem de iluminare este programat astfel încât să emită culori diferite în funcție de direcția deplasării robotului: lumina verde indică mișcarea înainte, albastru pentru viraj la stânga, mov pentru viraj la dreapta, iar roșu semnalează oprirea robotului. Această caracteristică nu numai că îmbunătățește aspectul estetic al robotului, dar servește și ca un instrument util pentru monitorizarea și interpretarea rapidă a comportamentului său, făcându-l mai interactiv și mai ușor de urmărit în timpul funcționării.

3. Implementare

Pentru implementarea caracteristicilor menționate ale robotului, am utilizat biblioteci specifice în codul sursă. Acestea includ <Adafruit_NeoPixel.h> și <avr/power.h> pentru controlul și gestionarea benzii LED Neopixel, și <Servo.h> pentru manipularea motoarelor.

În prima secțiune a codului nostru, ne concentrăm pe declararea variabilelor și a porturilor componente. Această etapă este crucială pentru stabilirea unei baze solide pentru funcționarea codului. Declarările includ:

1. **Porturile Componentelor**: Aici definim porturile de pe placa Arduino la care sunt conectați senzorii, butonul de pornire, motoarele și banda de LED-uri. Aceasta asigură o comunicare corectă între placa Arduino și diferitele componente hardware ale robotului.
2. **Variabilele**: Declarăm o serie de variabile care vor fi folosite pentru a stoca și manipula datele primite de la senzori, precum și pentru a controla comportamentul motoarelor și al benzii LED. Aceasta include variabile pentru starea senzorilor, viteza motoarelor, starea butonului și culorile LED-urilor.

Această structurare inițială este esențială pentru a asigura că programul este bine organizat și ușor de urmărit, facilitând astfel atât dezvoltarea ulterioară, cât și depanarea.

În cadrul proiectului nostru, secțiunea următoare reprezintă funcția **setup()** din cadrul codului Arduino, care este concepută să se execute o singură dată, la momentul inițializării sau al resetării plăcii Arduino. Această funcție are ca scop principal stabilirea și configurarea inițială a diferitelor componente hardware. În contextul specific al proiectului nostru, aceasta implică inițializarea pinilor corespunzători componentelor hardware ale robotului.

Funcția **StartMotor()**este concepută pentru a controla mișcarea unui motor într-o manieră specifică. Aceasta primește patru parametri: `m1`, `m2`, `forward` și `speed`.

1. **m1 și m2**: Acești parametri sunt folosiți pentru a specifica pinii la care motorul este conectat. Prin controlul acestor pini, funcția reglează modul în care motorul funcționează.
2. **forward**: Acest parametru este un indicator boolean (poate fi 0 sau 1) care determină direcția de rotație a motorului.
3. **speed**: Acest parametru controlează viteza motorului. Valoarea acestuia este ajustată prin modulare cu lățime de impuls (PWM - Pulse Width Modulation), permițând o variație fină a vitezei motorului.

Comportamentul funcției:

- Oprire: Dacă viteza este 0, motorul se oprește.
- Mișcare înainte: Dacă forward este adevărat (1), motorul se rotește într-o direcție, controlată prin aplicarea unui semnal PWM pe pinul m1.
- Mișcare înapoi: Dacă forward este fals (0), motorul se rotește în direcția opusă, cu semnal PWM aplicat pe pinul m2.

Prin utilizarea semnalului PWM, funcția permite un control fin al vitezei motorului, facilitând navigarea precisă a robotului.

Funcția **loop()** este partea esențială a programului Arduino, executată repetitiv după ce funcția setup() a fost inițial rulată. În cadrul acestei funcții, comportamentul robotului line follower este definit și gestionat.

Comportamentul funcției:

1. **Resetarea LED-urilor**: La începutul fiecărei iterații, banda de LED-uri este resetată pentru a pregăti afișarea noilor culori.
2. **Verificarea Butonului**: Se verifică dacă butonul de pe robot a fost apăsat. Dacă da, robotul este activat pentru a începe să urmărească linia.
3. **Logica de Urmărire a Liniei**: În funcție de starea senzorilor de linie (stânga și dreapta), robotul decide să meargă înainte, să vireze la dreapta, la stânga sau să se oprească. Aceste decizii sunt bazate pe următoarele condiții:
    - Ambii senzori nu detectează linia: Robotul merge înainte, LED-urile devin verzi.

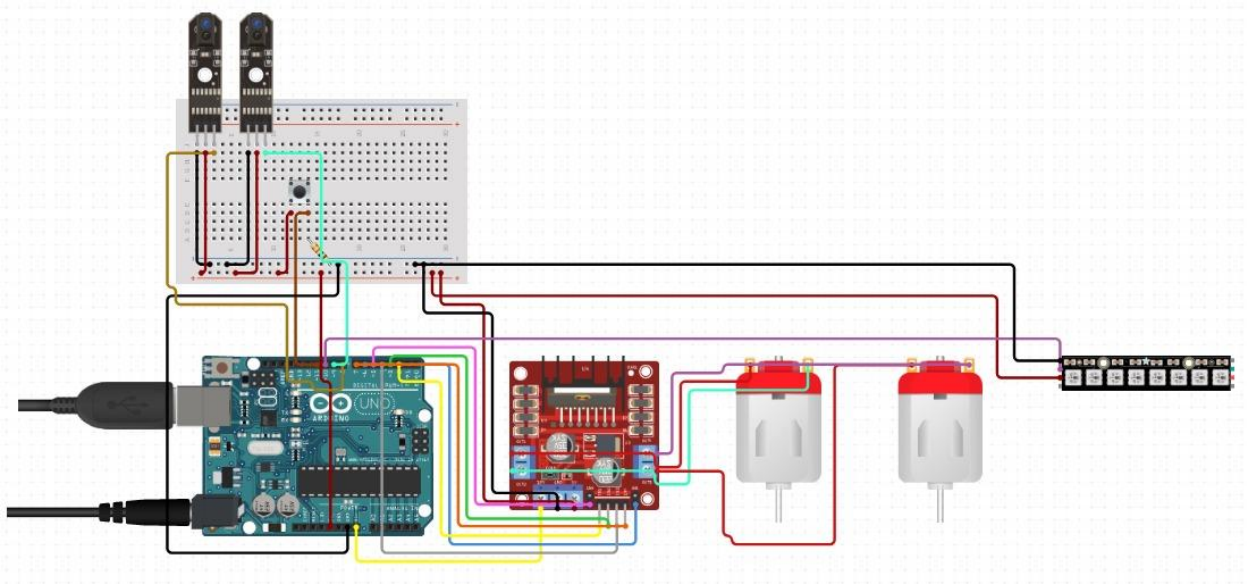
- Senzorul stâng detectează linia, dreptul nu: Robotul virează la dreapta, LED-urile devin albastre.
- Senzorul drept detectează linia, stângul nu: Robotul virează la stânga, LED-urile devin mov.
- Ambii senzori detectează linia: Robotul se oprește, LED-urile devin roșii.

4. **Controlul LED-urilor:** În fiecare situație, culoarea LED-urilor este actualizată pentru a reflecta mișcarea curentă a robotului, oferind un feedback vizual.
5. **Modul de Așteptare**: Dacă robotul nu este activat (butonul nu a fost apăsat), intră într-un mod de așteptare, indicat prin LED-uri roșii.

Funcția permite robotului să răspundă continuu și dinamic la schimbările detectate de senzori, asigurându-se că urmărește corect linia și reacționează corespunzător.

Celelalte funcții ale programului invocă metoda StartMotor în concordanță cu cerințele specifice de mișcare.

În cele ce urmează, vom prezenta diagrama de montaj:



4. Testare și validare

În etapa inițială a implementării proiectului, am întâmpinat o dificultate specifică: deși senzorii detectau corect linia și transmiteau informațiile necesare către motoare, am observat că motoarele reacționau cu o întârziere considerabilă. Inițial, am presupus că introducerea unor întârzieri (delays) în codul sursă, în momentul schimbării direcției de deplasare, ar atenua tranziția abruptă între diferitele stări ale motoarelor. Cu toate acestea, în urma unei analize mai aprofundate, am realizat că aceste întârzieri nu erau necesare. Motoarele erau suficient de robuste pentru a gestiona schimbările rapide de direcție fără riscul deteriorării. Prin eliminarea acestor întârzieri din cod, performanța robotului s-a îmbunătățit semnificativ, funcționând acum fără probleme. Această ajustare a fost un pas esențial în optimizarea răspunsului dinamic al robotului la stimuli și în îmbunătățirea eficienței generale a sistemului său de deplasare.

O provocare semnificativă în procesul de dezvoltare a fost calibrarea sensibilității senzorilor, astfel încât aceștia să identifice linia într-un mod optim. Ajustarea precisă a senzorilor infraroșu pentru a distinge eficient între diferitele suprafețe și a reacționa corespunzător la prezența liniei a fost esențială pentru funcționarea adecvată a robotului. Această etapă a necesitat un echilibru între sensibilitate și acuratețe, asigurându-se că senzorii sunt suficient de receptivi pentru a detecta linia, dar nu atât de sensibili încât să fie perturbați de variațiile minore ale suprafeței.

De asemenea, un alt aspect crucial în procesul de implementare a fost calibrarea vitezei motoarelor, o operațiune care a necesitat repetate ajustări. Am observat că, atunci când viteza de deplasare a robotului era excesiv de mare, senzorii nu reușeau să detecteze linia în timp util, rezultând în depășirea acesteia de către robot. Pe de altă parte, gestionarea curbelor a reprezentat o provocare distinctă: cu cât diferența de viteză între cele două motoare era mai mare, cu atât capacitatea robotului de a se realinia precis pe traseu era mai eficientă. Acest fapt a impus necesitatea unei echilibrări atente între viteza generală de deplasare și diferența de viteză între motoare pentru viraje, astfel încât robotul să urmărească linia cu acuratețe, fără a sacrifica stabilitatea sau reactivitatea sa. Prin urmare, setarea fină a vitezei motoarelor a fost un factor determinant în optimizarea performanței de urmărire a liniei de către robot.

5. Concluzie

Concluzionând, putem afirma că obiectivul stabilit pentru acest proiect a fost îndeplinit cu succes. Experiența dobândită și rezultatele obținute au deschis calea pentru noi direcții de îmbunătățire și inovație. În perspectiva viitoare, intenționăm să extindem capabilitățile robotului prin implementarea controlului la distanță, utilizând o telecomandă însoțită de un receptor infraroșu. Acest upgrade va oferi o interfață utilizator mai intuitivă și flexibilă pentru operațiuni precum pornirea și oprirea robotului.

În plus, planificăm să integrăm un senzor ultrasonic pentru detectarea obiectelor. Acesta va permite robotului să navigheze în medii mai complexe, evitând obstacolele și îmbunătățind siguranța și autonomia sa.

Alte îmbunătățiri pot include optimizări ale algoritmilor de urmărire a liniei, mărirea eficienței energetice și explorarea unor noi funcționalități care să extindă gama de aplicații ale robotului. Prin aceste dezvoltări, ne propunem să perfecționăm designul și funcționalitatea robotului, transformându-l într-un dispozitiv mai robust și versatil.