



UNIVERSITATEA DIN
BUCUREŞTI



FACULTATEA DE
MATEMATICA ŞI
INFORMATICA

SPECIALIZAREA TEHNOLOGIA INFORMAȚIEI

Lucrare de licență

MAȘINĂ CONTROLATĂ PRIN BLUETOOTH

Absolvent

Gabriel-Bogdan Angheloa

Coordonator științific

Lect. Univ. Dr. Alecsandru Chiroșca

București, iunie 2023

Rezumat

Considerând evoluția constantă a tehnologiei în zilele de astăzi, alături de felul firesc în care oamenii folosesc gesturi de mâna pentru a-și evidenția/clarifica intențiile, recunoașterea gesturilor de mâna este privită ca un element crucial în interacțiunea om-calculator. Scopul proiectului este de a dezvolta o aplicație practică și interactivă care oferă utilizatorului, pe lângă metodele tradiționale de a controla o mașină teleghidată, opțiunea de a manevra prin intermediul gesturilor captate de camera calculatorului.

Proiectul se bazează pe utilizarea de Arduino, un microcontroller puternic și versatil, care primește comenzi prin intermediul unei conexiuni Bluetooth. Pentru a detecta și interpreta gesturile utilizatorului, se utilizează camera calculatorului, în combinație cu limbajul de programare Python specializat în învățare automată. Algoritmii de recunoaștere a gesturilor de mâna sunt implementați în programul Python pentru a identifica gesturi, precum: „thumbs up”, „thumbs down”, „peace”, „rock” etc.. Pe baza acestor gesturi detectate de camera unității, comenziile corespunzătoare sunt transmise prin intermediul conexiunii Bluetooth de la computer către platforma Arduino, permitând mașinii să se deplaseze în direcția dorită.

Abstract

Considering the steady evolution of technology in our days, together with the natural way people use hand gestures to underline their intentions, hand gesture recognition is considered to be a crucial element in Human-Computer Interaction. The main scope of this project is to develop an application that is practical and intuitive which offers the user, besides the traditional means of controlling an RC Car, the option to control it through hand gestures recorded by a computer's camera.

This project is based on using the Arduino platform, a versatile and powerful microcontroller, which receives commands via a Bluetooth connection. In order to detect and interpret the user's hand gestures, the computer's camera will be used, together with a machine learning focused Python program. Hand gesture detection algorithms will be implemented in order to identify gestures, such as: "thumbs up", "thumbs down", "peace", "rock" etc.. Based on the detected hand gesture, the program sends the gesture's associated command, allowing the car to move in the desired direction.

Cuprins

1 Introducere	4
1.1 Motivație	4
1.2 Domenii abordate	4
1.3 Structura lucrării	5
1.3.1 Montarea și pregătirea mașinii teleghidate	5
1.3.2 Controlarea mașinii teleghidate printr-o aplicație mobilă	5
1.3.3 Implementarea comenziilor prin interpretarea gesturilor de mâna	6
2 Montarea și pregătirea mașinii teleghidate	7
2.1 Componentele necesare	7
2.1.1 Arduino Uno R3	8
2.1.2 Diagrama pinilor	9
2.1.3 Puntea H L298N	10
2.1.4 Modulul HC-05	11
2.1.5 Acumulatori 18650 si complementele lor	12
2.1.6 Motoarele DC	14
2.1.7 Șasiu	15
2.2 Configurarea conexiunilor între componente	16
2.2.1 Ansamblul general al sistemului implementat	16
2.2.2 Schema circuitului electric	16
2.3 Controlarea mașinii teleghidate	18
2.3.1 Sintaxa Arduino	19
2.3.2 Aplicația mobilă	25
2.3.3 Testarea mașinii	26
3 Implementarea comenziilor prin interpretarea gesturilor de mâna	27
3.1 Tehnologii utilizate	27
3.2 Sintaxa Python	28
4 Concluzii	35
4.1 Probleme întâmpinate și îmbunătățiri	35
Bibliografie	37

Capitolul 1

Introducere

1.1 Motivație

Datorită cursurilor și experiențelor din cadrul facultății, automobilismul, robotica și automatizarea proiectelor de tip DIY („Do It Yourself”) au devenit un subiect de mare interes pentru mine. Aceste domenii reprezintă o parte esențială din viitorul tehnologiei și pot aduce contribuții semnificative într-o multitudine de aspecte ale vieții cotidiene.

Prin crearea acestui proiect, scopul principal este de a construi ceva tangibil și a vedea conceptele și ideile materializate în ceva real, care să îmi încurajeze creativitatea, dezvoltându-mi atât abilitățile tehnice, cât și cele de soluționare a problemelor într-o manieră cât mai eficientă și pragmatică. Astfel, în această lucrare se vor aborda diferite provocări, care variază de la planificarea și montarea componentelor hardware până la programarea acestor componente pentru a obține un dispozitiv intuitiv, cât mai ușor de folosit de către utilizatori.

1.2 Domenii abordate

Acest proiect va include următoarele domenii principale:

- **Robotică**, folosită pentru a realiza, prin intermediul platformei Arduino Uno R3, conectarea și coordonarea componentelor necesare, creând astfel o mașină teleghidată.
- **Electronică**, folosită pentru a realiza integrarea electronică a tuturor componentelor într-o manieră corectă, fără a exista riscul apariției disfuncționalităților și erorilor în comunicare și control.
- **Inteligentă artificială**, folosită pentru a implementa tehnici de machine learning, prin care se pot recunoaște gesturile de mâna.

1.3 Structura lucrării

Această lucrare este împărțită în 3 capitole principale:

- **Montarea și pregătirea mașinii teleghidate:** Include prezentarea în detaliu a componentelor utilizate, configurarea și conectarea acestora într-un mod corect și coordonat.
- **Controlarea mașinii teleghidate printr-o aplicație mobilă:** Descrie procesul de creare a programului folosit de platforma Arduino și integrarea acestuia cu o aplicație mobilă, transformând telefonul într-un controller practic și portabil pentru mașina teleghidată.
- **Implementarea comenzi prin interpretarea gesturilor de mână:** Descrie procesul de implementare al algoritmului de machine learning care detectează în timp real gesturile de mână ale utilizatorului prin intermediul camerei computerului.

1.3.1 Montarea și pregătirea mașinii teleghidate

Acest pas se va realiza prin asamblarea și configurarea atentă a diferitelor componente. Microcontrollerul Arduino Uno R3 reprezintă nucleul sistemului, oferind funcționalitatea și controlul necesar. Folosind patru motoare de curent continuu, acestea vor fi direcționate și controlate cu ajutorul punții H L298N. Modulul Bluetooth HC-05 este conectat la Arduino și asigura comunicarea fără fir între mașină și dispozitivul de control. Ansamblul va beneficia de alimentare autonomă datorită bateriilor de tip 18650, acestea furnizând energia necesară pentru a asigura funcționarea optimă. Toate aceste componente vor fi montate pe un șasiu special conceput pentru prototipare, unde vom realiza conexiunile și fixările corespunzătoare pentru a obține o mașină teleghidată robustă și fiabilă.

1.3.2 Controlarea mașinii teleghidate printr-o aplicație mobilă

Acest pas va implica mai multe etape. Pentru început, vom crea un program dedicat pentru Arduino care va atribui și activa pinii specifici componentelor montate și îi va folosi pentru a controla motoarele și a le direcționa în funcție de comenzi primite. Funcțiile implementate în program permit mașinii să execute diferite deplasări, simple (înainte, înapoi, viraj/rotație stânga sau dreapta) sau compuse (înainte-stânga, înapoie-dreapta etc.).

Pentru a comunica cu aplicația mobilă, se va folosi modulul HC-05 pentru a realiza conexiunea între Arduino și telefonul mobil. Odată ce aplicația a realizat conexiunea, utilizatorul poate transmite comenzi de direcție care vor apela funcțiile menționate anterior, determinând mașina să execute mișcările dorite. Astfel, prin intermediul aplicației mobile, utilizatorul obține un control total asupra mașinii teleghidate și poate interacționa cu aceasta într-un mod convenabil și intuitiv.

1.3.3 Implementarea comenziilor prin interpretarea gesturilor de mâna

Acest pas se va realiza prin utilizarea limbajului de programare Python în combinație cu camera computerului și anumite biblioteci specializate în machine learning, în mod particular rețelele neuronale. Bibliotecile implementate oferă modele preantrenate pentru recunoașterea gesturilor de mâna, care, împreună cu captura de imagini realizată de cameră, pot determina, prin interpretare și predicție, gesturile recunoscute dintr-un set de clase predefinit.

Prin intermediul rețelelor neuronale și al modelor preantrenate, sistemul poate recunoaște gesturi specifice, precum „thumbs up”, „thumbs down”, „peace”, „rock”, „ok” și „stop”. În timpul funcționării, camera va captura imagini și va realiza predicții ale gesturilor de mâna în timp real. În funcție de ce gest este corelat cu ce comandă primește mașina, programul va transmite, prin intermediul tehnologiei Bluetooth, comenzi de deplasare ale mașinii dorite de către utilizator.

Astfel, prin intermediul acestei implementări bazate pe învățare automată și interpretarea gesturilor de mâna, mașina teleghidată poate fi controlată într-un mod intuitiv și interactiv.

Capitolul 2

Montarea și pregătirea mașinii teleghidate

2.1 Componentele necesare

Pentru realizarea unei mașini teleghidate este necesară o planificaremeticuloasă și o abordare logică în selecția componentelor adecvate. În procesul de selecție, este importantă considerarea unui buget rentabil astfel încât să se obțină rezultatele dorite fără compromiterea calității. Componentele selectate ar trebui să fie fiabile și să ofere performanță pe termen lung.

Analizând cu rigurozitate toate opțiunile disponibile, se va asigura o selecție optimă a componentelor, cu atenție acordată atât caracteristicilor tehnice, cât și costurilor. Integrarea și conectarea corespunzătoare a acestor componente vor conduce la dezvoltarea unei mașini teleghidate funcționale, scopul final al acestei abordări fiind obținerea unor rezultate constante care să ofere o soluție durabilă și de încredere în cadrul proiectului.

Proiectul de realizare a mașinii Arduino a fost un proces iterativ care a implicat evaluarea și selecția componentelor potrivite. În etapa inițială, am analizat piesele disponibile în diferite magazine de robotică și mecatronică pentru a identifica opțiunile ideale. Am luat în considerare aspecte precum calitatea, compatibilitatea și disponibilitatea pieselor.

Pe parcursul acestui proces, am făcut mai multe iterații, comparând caracteristicile și specificațiile diferitelor componente. Această abordare iterativă ne-a permis să obținem o combinație optimă de componente, care să îndeplinească cerințele proiectului. Am luat în considerare atât performanța tehnică, cât și fezabilitatea din punct de vedere al costurilor și disponibilității pieselor.

Astfel, am ajuns la o listă finală de piese necesare pentru construcția mașinii Arduino. Rezultatul final ne permite să construim mașina Arduino conform specificațiilor dorite și să obținem performanțele așteptate în funcționarea sa.

2.1.1 Arduino Uno R3

Arduino Uno R3[2] este un microcontroller versatil, utilizat în numeroase proiecte de robotică datorită funcționalităților sale și ușurinței de utilizare. Acest model, lansat în 2012 ca o evoluție a modelelor anterioare de Arduino Uno, a dobândit popularitate multumită flexibilității și extensibilității sale. Bazat pe microcontroller-ul ATmega328P[6], care oferă o putere mare de calcul pentru a rula programe complexe de control și monitorizare, această platformă oferă o gamă variată de porturi și pini de conectare, pe lângă interfețe USB și UART[28], care permit comunicarea și conectivitatea cu alte dispozitive.

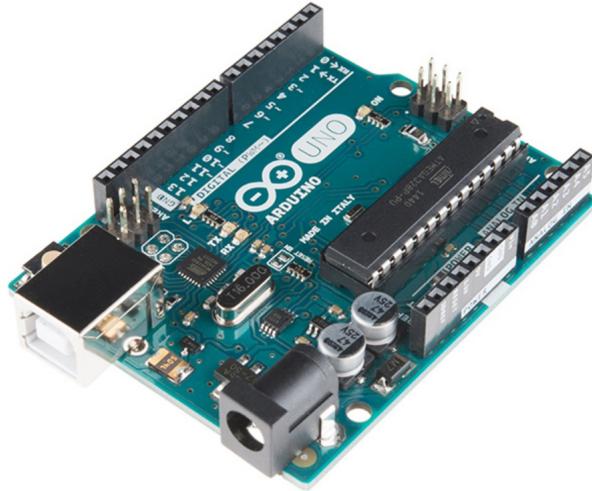


Figura 2.1: Microcontroller-ul Arduino Uno R3[10].

Un mare beneficiu al acestui microcontroller este modularitatea sa. Arduino, în prisma modularității sale, poate fi extins și personalizat prin intermediul unui ecosistem larg de componente suplimentare, cum ar fi senzori, module de comunicare și drivere de motor. Acest aspect permite realizarea unor proiecte complexe de robotică. De asemenea, în cazul în care procesorul Arduinoului se arde, acesta poate fi înlocuit relativ ușor datorită montajului pe bază de soclu, reducând complexitatea și costurile de reparație.

Toate aceste elemente, împreună cu numărul mare de resurse și o comunitate foarte activă, oferă posibilități de proiecte cu o capacitate de extindere considerabilă. Internetul oferă o vastă colecție de tutoriale, documentații și proiecte disponibile online care facilitează procesul de învățare și dezvoltare a aplicațiilor de robotică cu Arduino Uno R3.

2.1.2 Diagrama pinilor

Arduino Uno R3 este echipat cu o diagramă de pini[2] bine structurată, care oferă utilizatorilor o interfață convenabilă pentru conectarea și controlul diferitelor componente/periferice. Diagrama de pini a Arduino Uno R3 include următoarele elemente:

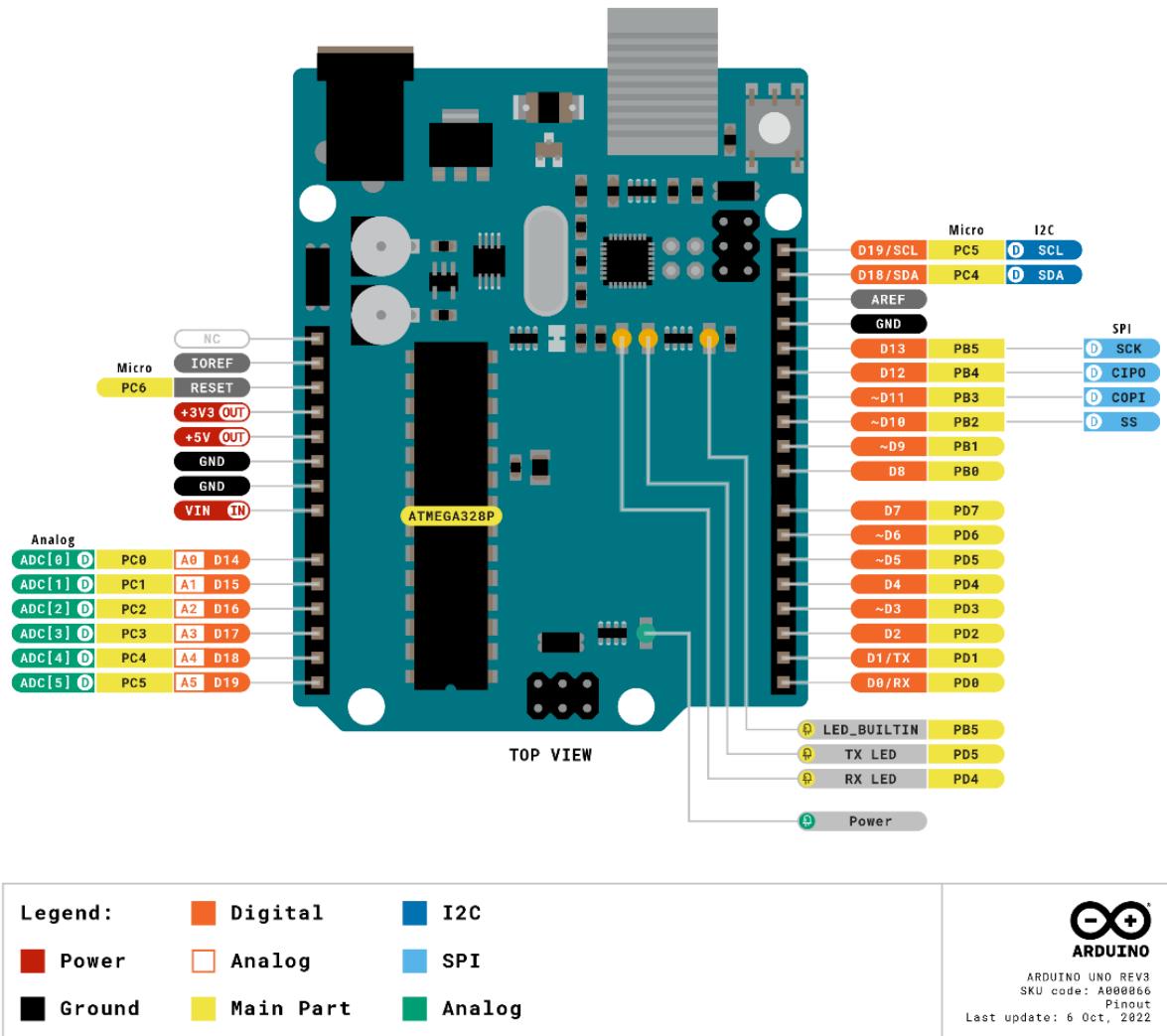


Figura 2.2: Diagrama pinilor pentru Arduino Uno R3[3].

- Pini digitali:** microcontroller-ul dispune de 14 pini digitali, numerotăți de la *D0* la *D13*. Acești pini pot fi utilizați pentru a citi și scrie semnale digitale. Pinii *D3*, *D5*, *D6*, *D9*, *D10* sunt, de asemenea, capabili să genereze semnale *PWM* (Pulse Width Module[19]). Acești pini vor fi cruciali pentru acest proiect, fiind folosiți pentru controlul precis al tensiunii de ieșire transmis la motoarele mașinii teleghidate. Astfel, putem controla viteza și direcția de deplasare într-un mod precis și adaptabil.

- **Pini analogici:** avem 6 pini analogici, numerotați de la *A0* până la *A5*. Acești pini pot fi utilizați pentru a măsura nivelurile de tensiune analogice, fiind utili pentru citirea valorilor senzorilor analogici(de proximitate, de lumină, de umiditate etc.).
- **Pini de alimentare:** Există mai mulți pini de alimentare pe placă (5V și 3.3V) care oferă tensiune constantă pentru alimentarea componentelor conectate. De asemenea, există și pini *GND* (masă) pentru a asigura conexiune închisă pentru circuitele conectate. Prezența pinului *VIN* (Voltage IN) oferă posibilitatea de a conecta mașina la o sursă de alimentare externă, care poate fi diferită de sursa de alimentare a calculatorului utilizat pentru programare. Prin conectarea mașinii la o sursă de curent externă prin pinul *VIN*, vom putea asigura o sursă de alimentare stabilă care susține funcționarea optimă a mașinii teleghidate.
- **Pini de comunicare serială:** Arduino Uno R3 dispune de pini de comunicare serială, *Rx* (Receive) și *Tx* (Transmit), care sunt utilizați pentru a comunica cu alte dispozitive, precum modulele Bluetooth sau portul serial al computerului.

2.1.3 Puntea H L298N

Puntea H L298N[5] este un circuit destinat controlării motoarelor de tip curent continuu. Unul dintre cele mai populare module care este utilizat în acest scop este modulul L298N. Aceasta este o punte H duală care poate controla până la 4 motoare DC(Direct Current[11]), sau două motoare DC separate și intermitent.

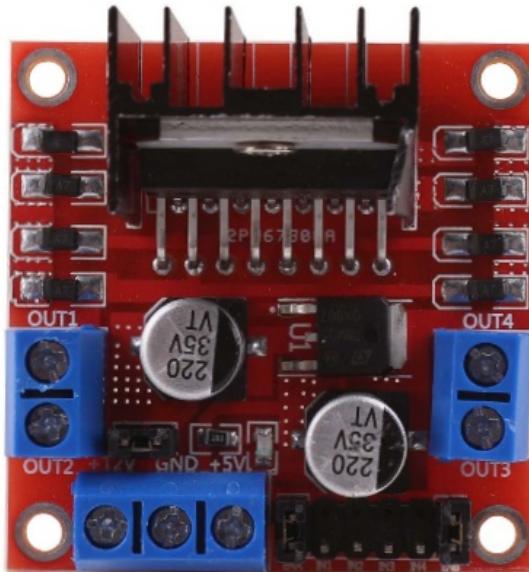


Figura 2.3: Modulul L298N[9].

Modulul L298N oferă o soluție convenabilă pentru gestionarea motoarelor în aplicații precum mașini teleghidate, fiind echipat cu patru pini de intrare (IN1 până la IN4) și patru pini de ieșire (OUT1 până la OUT4) care permit comandarea și inversarea direcției de roatare a motoarelor. Pentru acest proiect, este suficient să conectăm pinii motoarelor de pe partea stânga a mașinii la același set de pini (OUT1 și OUT2). Această conexiune în paralel ne permite să controlăm ambele motoare simultan.

Pe lângă funcția de control a direcției de roatare, putem controla și viteza de roatare a motoarelor prin utilizarea pinilor care oferă și PWM. În plus, acest modul este compatibil cu o plajă largă de tensiuni de alimentare, voltajul maxim fiind de 46V[8]. Pinul VIN va utiliza o sursă de curent externă pentru alimentarea modulului și a motoarelor. Astfel, modulul L298N devine o componentă cheie în construcția acestui proiect, rolul lui fiind de a controla direcția de roatare, viteza de roatare și alimentarea motoarelor.

2.1.4 Modulul HC-05

Modulul Bluetooth HC-05[7] este o componentă versatilă pentru proiecte din domeniul electronicii și roboticii, care permite comunicația wireless între dispozitive. În cazul acestui proiect, comunicația va avea loc între platforma Arduino și un laptop sau un telefon mobil în funcție de ce mod de control al mașinii se dorește a folosi. Conexiunea și configurarea modulului este realizată prin următorii pini:

- **Pinul VCC**, care este utilizat pentru alimentarea modului cu tensiune de 3.3-5V.
- **Pinul GND**, care este conectat la masă pentru a închide circuitul.
- **Pinii Tx și Rx**, care realizează comunicarea serială între modul și Arduino Uno R3.
- **LED-ul** montat pe modul va evidenția stadiul de conexiune, iar alături de pinul **STATE** se poate verifica pe post de feedback dacă dispozitivul funcționează în mod corect.
- **Butonul** este folosit pentru pinul de **ENABLE/KEY**, acesta comutând modulul în diferite moduri de comandă (slave - master).

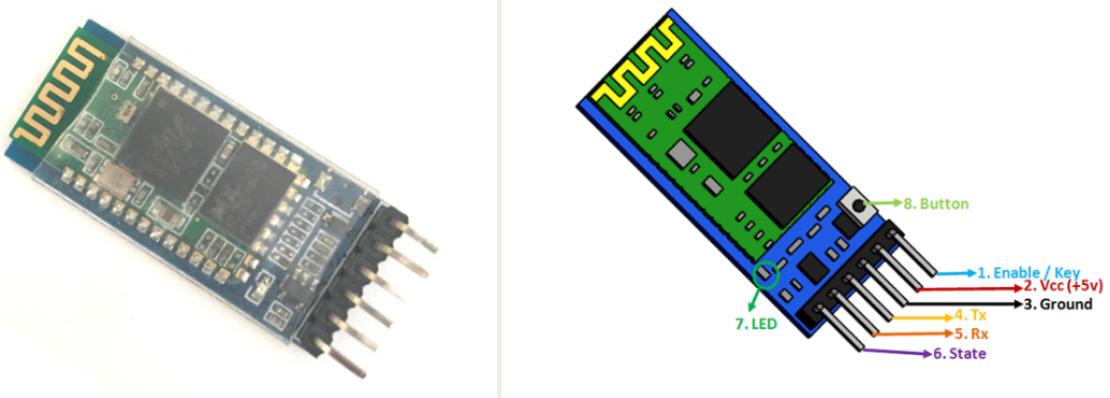


Figura 2.4: Modulul Bluetooth HC-05[7].

Un aspect deosebit de important care va fi abordat în utilizarea modulului HC-05 este pinul Rx. Acesta necesită o abordare specială deoarece este singurul pin care funcționează la o tensiune de 3.3V, în timp ce toate conexiunile de pe Arduino Uno R3 folosesc o tensiune de 5V. Pentru a evita potențiale probleme sau deteriorări ale modulului Bluetooth, va fi nevoie să implementăm un divizor de tensiune[32], un element care va fi prezentat în detaliu în momentul montării componentelor.

2.1.5 Acumulatori 18650 și complementele lor

Deși inițial ne-am gândit să folosim o baterie Li-Po, am constatat că această opțiune era costisitoare și implica pași adiționali pentru realizarea conexiunii, cum ar fi utilizarea de adaptoare pentru conectarea la modulul L298N.

Pe lângă costurile ridicate și pașii adiționali pentru conexiune, utilizarea unei baterii Li-Po(Litiu-Polimer[30]) în proiectul nostru ar fi implicat și riscuri suplimentare. Bateria Li-Po necesită o compartimentare specială și un regim de încărcare specific pentru a evita riscul de explozie sau incendiu. Având în vedere aceste aspecte, am căutat o alternativă mai sigură și mai convenabilă pentru alimentarea externă a mașinii Arduino. Astfel, am optat pentru acumulatorii 18650[24] ca soluție finală pentru alimentarea externă.

Bateriile de tip 18650 sunt acumulatori reincarcabili, denumirea lor venind de la dimensiunile standardizate: 18 milimetri în diametru, 65 milimetri în lungime, iar “50” indică formă lor cilindrică. Acești acumulatori folosesc tehnologia LI-Ion(Litiu-Ion[31]), care oferă o capacitate mare de stocare de energie. Pentru acest proiect, vom folosi modelul Panasonic NCR 18650, care ne oferă o capacitate de 3400 de mAh(milliamperi-oră[33]) la tensiunea nominală de 3.7 volți.

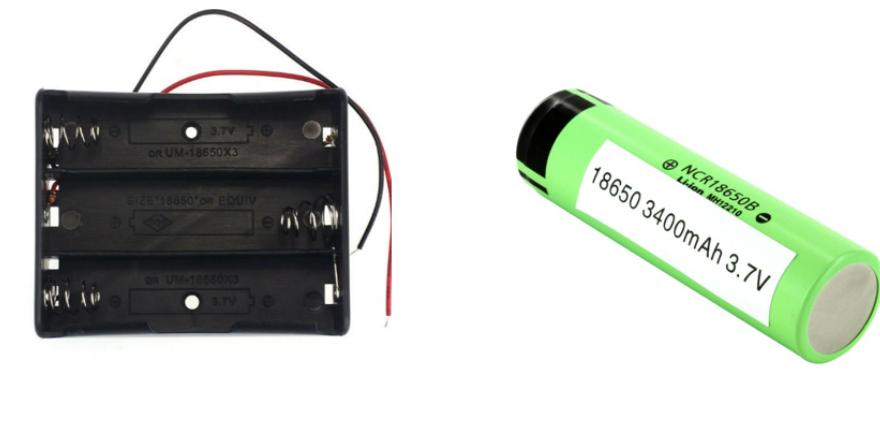


Figura 2.5: Acumulatorii 18650[16] și suportul de 3 acumulatori[23].

Am ales acest model de acumulator deoarece este considerat că fiind o “baterie de mare consum”[18] datorită fiabilității sale, duratei lungi de funcționare și capacitații sale de a fi reîncărcată de sute de ori. În plus, acestea au o adâncime mare de descărcare[18], ceea ce înseamnă că bateria poate fi descărcată complet și totuși poate fi complet reîncărcată (desi nu este recomandat).

Pentru a le implementa sub forma unei surse de curent externe în acest proiect, vom folosi un suport de 3 acumulatori 18650[23], alături de un întrerupător KCD1[4]. Scopul întrerupătorului este de a oferi utilizatorului controlul convenabil al alimentării, permitând oprirea și pornirea ușoară a alimentării cu energie a mașinii Arduino și a componentelor sale, în scopul economisirii energiei și prevenirea descărcării necontrolate a bateriilor.



Figura 2.6: Întrerupătorul KCD1[12].

Întrerupătorul KCD1 oferă o opțiune convenabilă de montare datorită pinterilor laterali, care permit instalarea sa pe una dintre plăcile de plastic ale șasiului cu ușurință. Această montaj necesită o mică prelucrare a plăcii, însă decupajele deja existente asigură o poziționare intuitivă. Prin decuparea unei mici secțiuni între două decupaje din placă, este posibilă montarea și fixarea întrerupătorului, oferind astfel un punct solid pentru a-l comuta și permitând utilizatorului să gestioneze alimentarea în mod convenabil.

2.1.6 Motoarele DC

Motorul DC cu cutie de viteze, cunoscut și sub denumirea de „motor TT”[15], este un motor electric compact, având o formă similară literei „T”. Acesta este compus dintr-un ax central cu două brațe laterale care conțin motorul și cutia de viteze. Acest motor este cunoscut deoarece este des folosit pentru aplicații în robotică datorită dimensiunii sale compacte și raportului calitate-preț foarte bun, fiind proiectat pentru a funcționa într-o plajă de tensiune între 3 și 6V, generând astfel până la 200 de rotații pe minut. În cazul acestui proiect, folosirea a 4 motoare de acest fel reprezintă soluția perfectă pentru controlarea precisă și relativ rapidă a mașinii teleghidate.

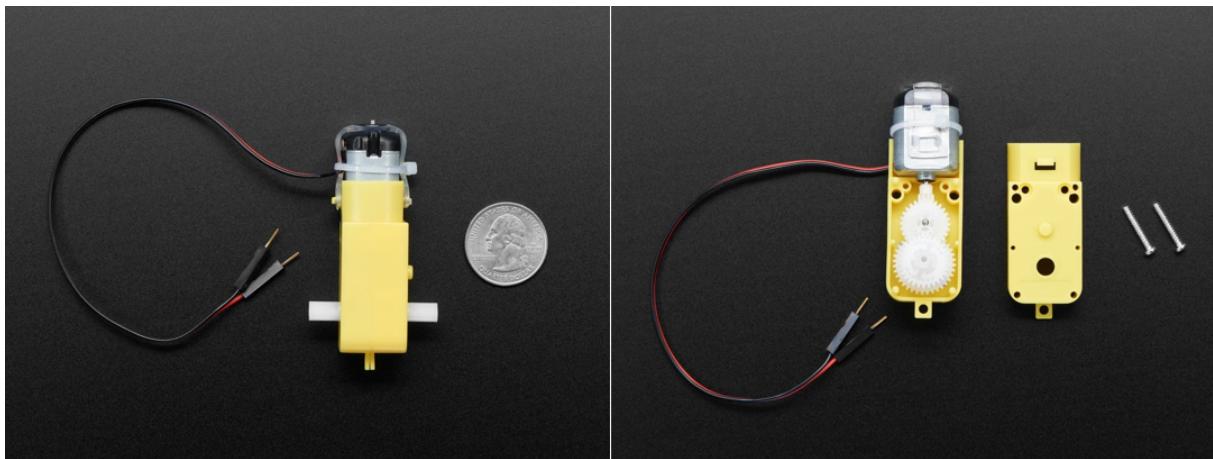


Figura 2.7: Structura motoarelor TT[15].

Utilizarea a patru motoare TT în construcția unei mașini Arduino cu tracțiune integrală oferă avantaje semnificative, în special în ceea ce privește controlul simplu și eficient al mașinii. Această configurare permite mașinii să se manevreze similar unui tanc pe senile. Prin controlul individual al fiecărui motor, mașina poate executa mișcări precise în diverse direcții, cum ar fi deplasarea înainte/înapoi, rotirea la stânga/dreapta și chiar virajele laterale. Astfel, mașina Arduino cu tracțiune integrală și patru motoare TT este potrivită pentru ceea ce dorim să se realizeze în acest proiect.

2.1.7 Șasiu

Șasiul mașinii Arduino reprezintă componenta finală și esențială asamblării acestui proiect, fiind platforma cheie pentru integrarea și asamblarea tuturor celorlalte piese menționate anterior. Acesta este compus din două plăci de plastic, prevăzute cu multiple decupaje care permit montarea și conectarea componentelor într-un număr mare de configurații. Pentru fixarea motoarelor TT și asamblarea celor două plăci, se vor folosi șuruburi, piulițe și distanțiere. Prin intermediul acestui șasiu, toate componentele pot fi integrate într-un mod organizat, oferind posibilitatea de a adapta și personaliza proiectul în funcție de posibilele modificări sau îmbunătățiri realizate de-a lungul acestei lucrări.



Figura 2.8: Șasiul asamblat împreună cu motarele TT[1].

O problemă întâmpinată a fost legată de calitatea componentelor. În cazul motoarelor TT și al roțiilor incluse cu șasiul, deși oferă performanțe bune în raport cu prețul lor, aspectul calității a lăsat mult de dorit. Atât roțile, cât și axele motoarelor erau confecționate din plastic, iar introducerea roțiilor pe axe era imposibilă din cauza lipsei de toleranță între ax și motor. Pentru a rezolva această problemă, a fost necesară prelucrarea manuală a axelor, subtiându-le cu ajutorul unui șmirghel abraziv.

2.2 Configurarea conexiunilor între componente

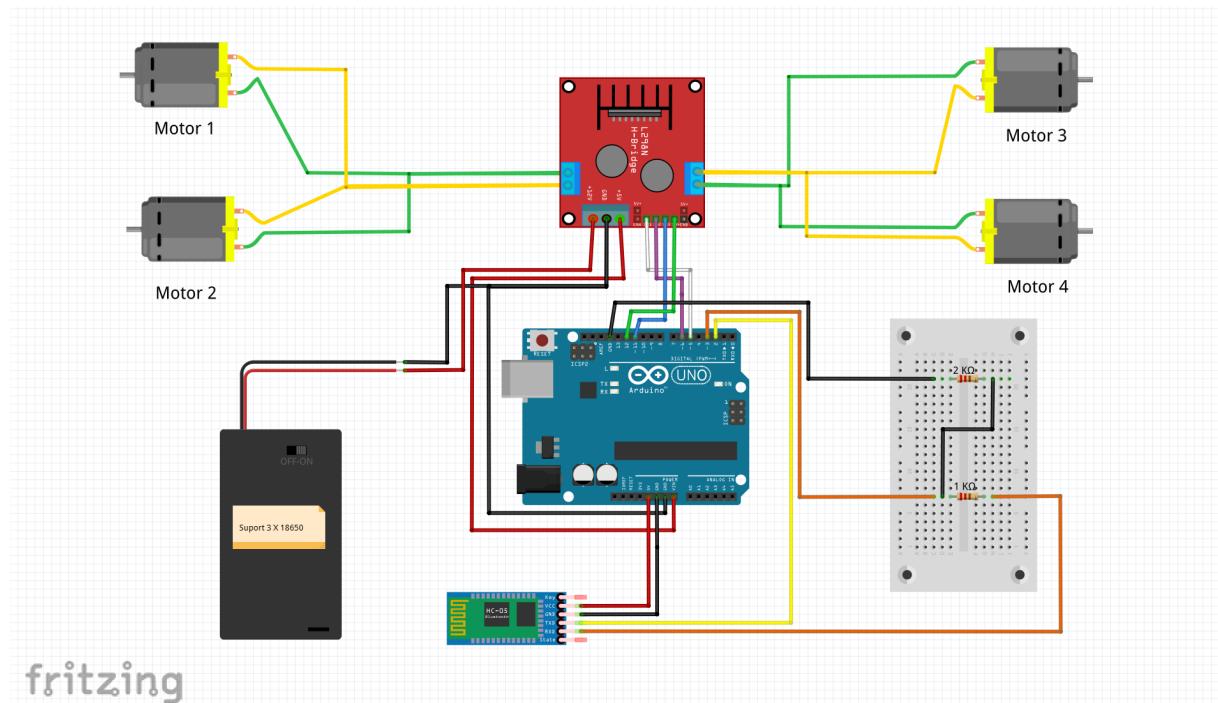
2.2.1 Ansamblul general al sistemului implementat

Pentru a plănuia și vizualiza într-un mod pragmatic toate elementele care vor forma mașina teleghidată, vom folosi Fritzing[29], un program open-source, destinat construirii de arhitecturi hardware în urma prototipării. Acesta a fost dezvoltat de către un grup de cercetători de la Universitatea Postdam din Germania în 2007 și, de atunci, a devenit o unealtă esențială pentru dezvoltatori în realizarea de proiecte electronice. Fritzing oferă o interfață intuitivă, care permite crearea de diagrame schematiche și de plăci de circuite imprimate(Printed Circuit Boards[22]) pentru prototiparea digitală a acestui proiect.

Cu ajutorul bibliotecii extinse de componente electronice și module Arduino disponibile, va fi posibilă implementarea și conectarea tuturor componentelor considerate pentru realizarea mașinii Arduino. Astfel, vom vizualiza și verifica conexiunile și interacțiunile între componente, testând și validând proiectul înainte de a trece la etapa de construcție fizică.

2.2.2 Schema circuitului electric

Asamblarea și conexiunile vor fi ilustrate prin prezentarea relației dintre o componentă centrală și seturi de componente asociate.



Motoarele și puntea L298N

Așa cum am menționat anterior, puntea H utilizată în proiect este proiectată inițial pentru controlul sensului de rotație și a vitezei a maximum două motoare DC, în timp ce proiectul nostru implică utilizarea a patru motoare TT. Pentru a depăși această limită hardware, motoarele situate pe partea stângă și pe partea dreaptă a șasiului vor fi conectate în paralel.

În diagrama circuitului, motoarele 1 și 2 sau motoarele 3 și 4 vor fi tratate ca un singur motor. Firul de tensiune (galben) al motorului 1 va fi conectat împreună cu firul de tensiune al motorului 2 la pinul *OUT2* al punții L298N, în timp ce motoarele 3 și 4 vor utiliza pinul *OUT4*. În ceea ce privește firele de masă, motoarele 1 și 2 vor fi conectate împreună la pinul *OUT1*, iar pentru motoarele 3 și 4 se va utiliza pinul *OUT3*.

Arduino și puntea L298N

Conexiunea între aceste componente este crucială deoarece microcontroller-ul Arduino va controla modulul L298N și implicit motoarele TT. În fază inițială, microcontroller-ul va fi alimentat și controlat prin cablul USB conectat la calculator pentru a testa corectitudinea funcțiilor de deplasare.

Partea de control a motoarelor TT se realizează prin conectarea pinilor 5, 6, 11 și 12 de pe microcontroller la pinii *IN1*, *IN2*, *IN3* și *IN4* ai punții H. Acești pini au fost aleși de pe Arduino Uno R3 deoarece permit utilizarea modulației în lățime de impuls (PWM), care ne permite să trimitem semnale ce pot simula semnale analogice. Prin selectarea valorilor dintr-o plajă între 0 și 255, vom putea genera diferite viteze de rotație pentru motoarele TT.

Pentru alimentare, va fi suficient momentan să utilizăm pinul *VIN* și unul din pinii *GND* de pe Arduino. Pinul *VIN* va fi conectat la pinul *+5V*, iar pinul *GND* va fi conectat la pinul *GND* al modulului L298N.

Acumulatorii cu întrerupător și puntea L298N

Acumulatorii și întrerupătorul sunt elemente esențiale pentru controlul alimentării și prevenirea descărcării inutile în cazul în care mașina teleghidată nu este utilizată. Conexiunea este simplă: firul de tensiune (roșu) va fi conectat la pinul *+12V*, iar firul de masă (negru) va fi conectat la pinul *GND* al modulului L298N. Este important de menționat că pinul *GND* al punții H va avea acum două fire conectate la el, unul provenind de la acumulatori și celălalt de la microcontroller. Astfel, vom obține un circuit care furnizează o sursă de alimentare externă, permitând platformei Arduino să funcționeze fără a mai fi necesar cablul USB de alimentare.

Arduino și modulul HC-05

Conexiunea dintre modulul HC-05 și Arduino Uno R3 ne va permite controlul de la distanță al mașinii teleghidate. Pinii *VCC* și *GND* ai modulului vor fi conectați la pinul de *5V*, respectiv, alt pin de *GND* liber de pe Arduino, iar pinul *Tx* (transmisie) îl vom conecta la pinul *0* de pe microcontroller.

Pinul *Rx* va necesita atenție adițională, deoarece acesta funcționează la tensiunea de *3.3V*, comparativ cu restul pinilor de pe Arduino R3 care utilizează o tensiune de *5V*. Pentru a evita deterioararea modulului, vom implementa un divizor de tensiune, un circuit simplu format din două rezistoare conectate în serie. În cazul acestui proiect, vom folosi un rezistor de *1 KΩ* și unul de doi *KΩ*. Firul legat cu cel de doi *KΩ* va fi conectat la unul din pinii *GND* rămași liberi, iar cel legat de rezistență de un *KΩ* va fi conectat la pinul *Tx*. Valorile rezistențelor au fost determinate folosind următoarea formulă:

$$V_{out} = \frac{V_s \times R_2}{(R_1 + R_2)}$$

Unde:

- V_{out} reprezintă tensiunea de ieșire.
- V_s reprezintă tensiunea de intrare.
- R_1 și R_2 reprezintă rezistențele folosite, R_1 fiind cel de *1 KΩ*, iar R_2 fiind cel de doi *KΩ*.

Aditia acestui divizor de tensiune va reduce nivelul de tensiune de la *5V* al pinului *Tx* al platformei Arduino la *3.3V*, valoarea optimă pentru funcționarea corectă a modulului HC-05.

2.3 Controlarea mașinii teleghidate

Varianta în care mașina teleghidată este controlată folosind un telefon mobil oferă o metodă tradițională de control, asemenea unui controlor. Prin intermediul unei aplicații mobile special create în scop educațional, utilizatorul poate trimite comenzi către Arduino pentru a controla mișările mașinii. Codul încărcat în Arduino este programat să interpreteze aceste comenzi și să acționeze în consecință, permitând mașinii să efectueze diverse manevre. Fiecare tip de manevră, cum ar fi deplasarea înainte, înapoi, în stânga sau în dreapta, este asociat cu o anumită comandă trimisă de aplicația mobilă. Astfel, utilizatorul poate experimenta și înțelege principiile de bază ale controlului mașinii teleghidate.

2.3.1 Sintaxa Arduino

Limbajul de programare folosit pentru dezvoltarea programelor Arduino este un dialect al limbajului C/C++[21]. Structura sintaxei urmează o structură logică și specifică, fiind compusă din diferite elemente esențiale:

- **Funcția setup():**

Funcția *setup()* este prima funcție care se execută într-un program Arduino. Aceasta este utilizată pentru a configura setările și inițializările necesare pentru proiect, rulând o singură dată. În interiorul funcției *setup()*, se pot realiza acțiuni precum setarea pinilor ca intrări sau ieșiri, initializarea comunicării seriale și alte configurații specifice proiectului.

- **Funcția loop():**

Funcția *loop()* este funcția principală a programului Arduino și se execută în mod repetat într-o buclă infinită. Toate instrucțiunile scrise în interiorul acestei funcții vor fi executate în mod continuu. În funcția *loop()*, se pot programa acțiuni specifice, cum ar fi citirea valorilor de la senzori, controlul dispozitivelor conectate și alte operațiuni care trebuie să se desfășoare în mod continuu pe durata funcționării programului.

- **Instrucțiuni și funcții:**

În interiorul funcțiilor *setup()* și *loop()*, pot fi utilizate diferite instrucțiuni și funcții necesare pentru acest proiect. Acestea includ instrucțiuni de control precum *if*, *for*, *while*, instrucțiuni de atribuire a valorilor, apeluri de funcții predefinite și funcții definite de utilizator. Aceste instrucțiuni și funcții sunt utilizate pentru a realiza sarcinile specifice proiectului și pentru a controla dispozitivele și senzorii conectați la placa Arduino.

Pentru a crea un program care ne oferă un context cât mai clar în timpul realizării programului, vom declara următoarele constante și variabile:

```
1 #define IN1 5 // Left motors, forward
2 #define IN2 6 // Left motors, backward
3 #define IN3 11 // Right motors, forward
4 #define IN4 12 // Right motors, backward
5
6 int command;
7 int speed = 255; // 0 - 255.
8 int speedSec;
9 int turnRadius = 15;
```

Constantele simbolice *IN1*, *IN2*, *IN3* și *IN4* au atribuite valorile pinului folosit pentru a realiza conexiunea cu placa Arduino. În funcție de convenția decisă pentru ce direcție este considerată înainte sau înapoi a mașinii Arduino, am stabilit ca *IN1* și *IN2* să reprezinte motoarele de pe partea stânga a șasiului, cu direcția de deplasare înainte, respectiv înapoi. Consecvent, constantele *IN3* și *IN4* reprezintă motoarele de pe partea dreaptă a șasiului, cu direcția de deplasare înainte, respectiv înapoi.

Următoarele variabile declarate sunt variabile care pot reține valori numerice întregi. Variabila *command* este variabila care preia comenzi destinate deplasării mașinii și *speed* reține viteza de rotație trimisă ca semnal PWM motoarelor TT. *speedSec* și *turnRadius* vor fi folosite împreună pentru a stabili viteza de rotație a motoarelor aflate pe partea interioară a unui viraj.

De exemplu, în cazul în care mașina se va deplasa înainte, virând spre dreapta, motoarele TT de pe partea stânga a șasiului vor primi ca viteză de rotație valoarea salvată în variabila *speed*, iar motoarele de pe partea dreapta vor primi ca viteză de rotație valoarea salvată în variabila *speedSec*. Diferența de viteză dintre cele două punți laterale vor face ca mașina să vireze spre dreapta. Astfel, mașina se va deplasa într-un mod similar unui vehicul cu șenile, virajele fiind realizate prin diferența de rotație sau sens ale punților laterale.

```

1 void setup() {
2     pinMode(IN1, OUTPUT);
3     pinMode(IN2, OUTPUT);
4     pinMode(IN3, OUTPUT);
5     pinMode(IN4, OUTPUT);
6     Serial.begin(9600); //Set the baud rate to your Bluetooth module.
7 }
```

Functia *setup()* pregătește mediul de lucru prin configurarea conexiunilor și comunicațiilor necesare pentru funcționarea corectă a mașinii Arduino. Prin apelul funcției *pinMode()*, se setează modul de funcționare al pinilor *IN1*, *IN2*, *IN3*, *IN4* ca ieșiri digitale pentru a controla conexiunile modulului L298N și pentru a furniza semnale de ieșire către motoarele mașinii.

Apelul funcției *Serial.begin(9600)* configurează comunicarea serială și rata de transfer, cunoscută și sub denumirea de *Baud Rate*[13], care reprezintă numărul de biți pe secundă transmiși sau recepționați între dispozitivele care comunică între ele. În cadrul acestui proiect, vom seta rata de transfer la 9600 de biți pe secundă pentru a realiza comunicarea cu modulul HC-05, care este utilizat pentru conectivitatea Bluetooth a mașinii Arduino.

```

1 void loop() {
2     if (Serial.available() > 1) {
3         command = Serial.read();
4         stop(); // Initialize with motors stoped.
5         switch (command) {
6             case 'F':
7                 forward();
8                 break;
9             case 'B':
10                back();
11                break;
12             case 'L':
13                 left();
14                 break;
15             case 'R':
16                 right();
17                 break;
18             case 'G':
19                 forwardleft();
20                 break;
21             case 'I':
22                 forwardright();
23                 break;
24             case 'H':
25                 backleft();
26                 break;
27             case 'J':
28                 backright();
29                 break;

```

Funcția *loop()* reprezintă bucla principală de execuție a programului Arduino. Aceasta se execută continuu, repetându-se în mod ciclic până când alimentarea microcontroller-ului este întreruptă. Apelul funcției *Serial.available()* verifică disponibilitatea datelor seriale, returnând numărul de bytes disponibili pentru citire din buffer-ul serial. În cazul în care valoarea returnată este mai mare decât 1, se citesc datele seriale folosind funcția *Serial.read()* și se stochează în variabila *command*.

Odată ce a fost citită prima comandă, se va apela funcția *Stop()* pentru a inițializa motoarele. Restul comenziilor care vor fi primite prin datele seriale vor fi filtrate printr-o structură de control *switch-case*, pentru a executa acțiuni specifice în funcție de valoarea comenzi citite. Fiecare caz reprezintă o comandă posibilă și corespunde unei funcții specifice de mișcare. În plus, au fost considerate cazuri pentru setarea vitezei de mișcare.

```

1   case '0':
2       Speed = 100;
3       break;
4   case '1':
5       Speed = 140;
6       break;
7   case '2':
8       Speed = 153;
9       break;
10  case '3':
11      Speed = 165;
12      break;
13  case '4':
14      Speed = 178;
15      break;
16  case '5':
17      Speed = 191;
18      break;
19  case '6':
20      Speed = 204;
21      break;
22  case '7':
23      Speed = 216;
24      break;
25  case '8':
26      Speed = 229;
27      break;
28  case '9':
29      Speed = 242;
30      break;
31  case 'q':
32      Speed = 255;
33      break;
34 }
35 speedSec = turnRadius;
36 }
37 }
```

Comenziile care sunt cifre între 0 și 9 vor actualiza variabila *speed*, iar în cazul comenzi *q*, viteza va fi setată la valoarea maximă. În funcție de valoarea atribuită vitezei de rotație, variabila *speedSec* va fi modificată în funcție de *turnRadius*.

Următoarele funcții sunt responsabile de controlul motoarelor mașinii Arduino în diferite direcții și la diferite viteze. Ele utilizează funcția *analogWrite()* pentru a trimite semnale de control către pinii conectați la modulul L298N.

```
1 void forward() {
2     analogWrite(IN1, speed);
3     analogWrite(IN3, speed);
4 }
5
6 void back() {
7     analogWrite(IN2, speed);
8     analogWrite(IN4, speed);
9 }
```

- **void forward():** Această funcție este responsabilă de deplasarea înainte a mașinii. Se setează valoarea de viteză *speed* pe pinii *IN1* și *IN3*, ceea ce va duce la mișcarea înainte a motoarelor.
- **void back():** Această funcție realizează mișcarea înapoi a mașinii. Valoarea de viteză *speed* este setată pe pinii *IN2* și *IN4*, astfel încât motoarele să se rotească în direcția opusă, determinând mișcarea înapoi.

```
1 void left() {
2     analogWrite(IN4, speed);
3     analogWrite(IN1, speed);
4 }
5
6 void right() {
7     analogWrite(IN3, speed);
8     analogWrite(IN2, speed);
9 }
```

- **void left():** Această funcție controlează virajul la stânga al mașinii. Prin aceasta, motoarele se vor roti în sensuri opuse, ceea ce va duce la virajul la stânga al mașinii.
- **void right():** Această funcție controlează virajul la dreapta al mașinii. Similar cu funcția *left()*, se schimbă valorile de viteză Speed între pinii *IN3* și *IN2*, respectiv *IN4* și *IN3*. Astfel, motoarele vor avea rotații în sensuri opuse, ceea ce va determina virajul la dreapta al mașinii.

```

1 void right() {
2     analogWrite(IN3, speed);
3     analogWrite(IN2, speed);
4 }
5 void forwardleft() {
6     analogWrite(IN1, speed);
7     analogWrite(IN3, speedSec);
8 }
9 void forwardright() {
10    analogWrite(IN1, speedSec);
11    analogWrite(IN3, speed);
12 }
13 void backright() {
14     analogWrite(IN2, speedSec);
15     analogWrite(IN4, speed);
16 }
17 void backleft() {
18     analogWrite(IN2, speed);
19     analogWrite(IN4, speedSec);
20 }

```

- **void forwardleft(), void forwardright(), void backright(), void backleft():**

Aceste funcții controlează mișcarea mașinii în direcții diagonale. Ele combină diferite combinații de pini de control, *IN1*, *IN2*, *IN3* și *IN4* și setează valorile de viteză *speed* și *speedSec* pentru a obține mișcări specifice.

```

1 void stop() {
2     analogWrite(IN1, 0);
3     analogWrite(IN2, 0);
4     analogWrite(IN3, 0);
5     analogWrite(IN4, 0);
6 }

```

- **void Stop():** Această funcție oprește toate motoarele mașinii prin setarea valorii de viteză a pinilor *IN1*, *IN2*, *IN3* și *IN4* la 0. Aceasta va opri motoarele și va determina oprirea mașinii.

2.3.2 Aplicația mobilă

Aplicația “Bluetooth RC Controller”[26], disponibilă pe Play Store, este o opțiune ideală pentru acest proiect din mai multe motive:

- **Conecțivitate Bluetooth:**

Aplicația utilizează tehnologia Bluetooth pentru a permite controlul dispozitivelor prin intermediul unui dispozitiv mobil. Arduino va folosi modulul Bluetooth HC-05 și va facilita comunicarea fără fir între Arduino și dispozitivul mobil.

- **Personalizare:**

Aplicația permite personalizarea mașinii Arduino în funcție de preferințele utilizatorului. Utilizatorul poate configura diferite comenzi pentru mișcare înainte, înapoi, viraje la stânga, viraje la dreapta și alte funcții specifice. De asemenea, pot exista opțiuni pentru ajustarea vitezei și accelerării mașinii.

- **Ușurință utilizării:**

Interfața aplicației este proiectată pentru a fi intuitivă și ușor de utilizat, astfel încât chiar și utilizatorii mai puțin experimentați să poată controla mașina Arduino fără probleme. Utilizatorii pot naviga și pot înțelege rapid modul de utilizare.

- **Compatibilitate:**

Aplicația poate fi compatibilă cu o gamă largă de dispozitive mobile, inclusiv smartphone-uri și tablete care rulează sistemul de operare Android. Aceasta oferă o flexibilitate crescută și oportunitatea de a utiliza dispozitivele mobile existente pentru a controla mașina Arduino.



Figura 2.10: Interfața aplicației mobile[26].

2.3.3 Testarea mașinii

Odată ce am trecut prin etapa de asamblare, programare și conectare dintre dispozitive, ne concentrăm pe evaluarea performanțelor mașinii Arduino. Testarea reprezintă un moment crucial în dezvoltarea acestui proiect, oferindu-ne oportunitatea de a evalua rezultatele obținute și de a face ajustări necesare pentru a îmbunătăți performanțele și experiența utilizatorului.

Prin implementarea celor trei baterii 18650, am reușit să generăm suficientă putere pentru ca mașina să atingă viteze maxime surprinzătoare de bune. De asemenea, modulul Bluetooth HC-05 realizează o conexiune stabilă între mașină și telefonul utilizatorului, permitând controlul mașinii de la distanțe de până la 10 metri. Această capacitate de control la distanță extinde versatilitatea, utilizarea mașinii Arduino fiind posibilă într-o arie de operațiune mult mai mare.

Un aspect remarcabil al mașinii Arduino este capacitatea sa de a fi manevrată cu ușurință și precizie în timpul rulării. Această caracteristică este posibilă datorită abilității utilizatorului de a ajusta diferența de viteză între motoarele punților laterale. Prin implementarea variabilei *turnRadius* și gestionarea acesteia în cadrul structurii de control *switch-case*, mașina devine extrem de manevrabilă, permitând realizarea de viraje și rotații prompte și precise.

Cu toate că în prezent mașina se prezintă ca un prototip și necesită o carcasă pentru ascunderea componentelor, ansamblul demonstrează o rezistență și robustețe remarcabilă. Această caracteristică esențială permite mașinii să reziste impacturilor cauzate de posibile erori de manevrare și să continue să funcționeze într-un mod fiabil și eficient. Cu fiecare testare, s-au adunat informații valoroase și s-au identificat posibilele îmbunătățiri pentru a se asigura că mașina Arduino va atinge performanțe și rezultate cât mai promițătoare.

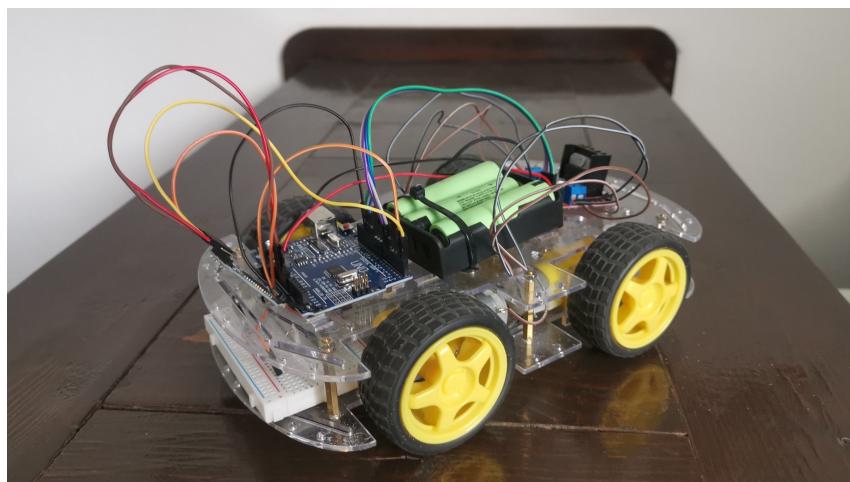


Figura 2.11: Mașina Arduino în stadiul de teste.

Capitolul 3

Implementarea comenziilor prin interpretarea gesturilor de mână

Implementarea recunoașterii de gesturi de mână și transmiterea comenziilor în funcție de acestea reprezintă următorul pas în evoluția proiectului, odată ce mașina Arduino a ajuns complet funcțională folosind aplicația mobilă. Vom aborda concepte de automatizare, utilizând imagini captate în timp real de camera unui laptop pentru a detecta diferite gesturi de mână și transmitând comenzi de deplasare corespunzătoare mașinii Arduino în funcție de acestea.

Utilizatorul va putea controla mișcările mașinii prin simpla folosire a gesturilor de mână, fără a mai fi nevoie de interacțiunea cu aplicația mobilă. Astfel, se va aborda o nouă perspectivă asupra posibilităților de control și de interacțiune cu dispozitivul, oferind o experiență și mult mai captivantă.

3.1 Tehnologii utilizate

Pentru a realiza programul pentru detectarea gesturilor de mâna vom folosi următoarele tehnologii:

- **Python:**

Limbajul de programare Python[25] este alegerea ideală pentru aplicațiile care utilizează inteligență artificială. Caracterizat prin simplitatea și lizibilitatea sa, sintaxa clară a limbajului Python face codul ușor de înțeles și de adaptat, acestea fiind niște aspecte cruciale în dezvoltarea modelelor de ML. În plus, Python oferă un ecosistem vast de biblioteci și framework-uri specializate, ce vor facilita dezvoltarea și antrenarea rețelelor neuronale[14].

- **OpenCV:**

OpenCV[20] (Open Source Computer Vision Library) este un framework folosit pentru vederea artificială și învățarea automată a calculatorului. Acest software open-source conține peste 2500 de algoritmi optimizați, ce pot detecta și recunoaște fețe, identifica obiecte, clasifica gesturi faciale sau de mâna din videoclipuri, urmări obiecte aflate în mișcare etc. În cazul acestui proiect, OpenCV va fi folosit pentru detectarea gesturilor de mâna.

- **TensorFlow:**

TensorFlow[27] este una dintre cele mai populare și folosite biblioteci de învățare automată disponibile la momentul actual. Dezvoltat de Google Brain, acesta a fost proiectat pentru a facilita dezvoltarea și implementarea modelelor de învățare automată, oferind posibilitatea de a antrena rețele neuronale în vederea distingerea a diferite gesturi de mâna.

3.2 Sintaxa Python

Pentru a realiza programul dorit, va fi nevoie să instalăm și să importăm toate bibliotecile necesare pentru detecția gesturilor de mâna.

```
1 import serial
2 import cv2
3 import numpy as np
4 import mediapipe as mp
5 import tensorflow as tf
6 from tensorflow import keras
7 from keras.layers import Dense
8 from keras.models import Sequential, load_model
```

Secvența de cod prezintă importul următoarelor biblioteci:

- **serial:** Această linie importă biblioteca *serial*, care este folosită pentru comunicarea serială între calculator și dispozitive externe, în cazul nostru, mașina Arduino.
- **cv2:** Acest import va oferi funcționalitățile bibliotecii *OpenCV* în program, oferind o colecție de algoritmi și instrumente pentru viziunea computerizată și prelucrarea imaginilor.

- **numpy**: Biblioteca *NumPy* va fi utilizată pentru a efectua operații matematice pe matrice și vectori multidimensionali.
- **mediapipe**: Biblioteca *MediaPipe* oferă un set de soluții pentru analiza și manipularea datelor media, inclusiv recunoașterea mâinilor.
- **tensorflow**: Biblioteca *TensorFlow* va permite antrenarea și utilizarea rețelelor neuronale.
- **from tensorflow import keras**: Această linie permite accesul la funcționalitățile bibliotecii *Keras*[17], care este o interfață de nivel înalt destinată construirii și antrenării modelelor de învățare automată în *TensorFlow*.
- **from keras.layers import Dense**: Această linie aduce clasa *Dense* din *Keras*, care este utilizată pentru a defini straturile complet conectate ale rețelelor neurale.
- **from keras.models import Sequential, loadmodel**: Această linie aduce clasele *Sequential* și *load_model* din *Keras*. Clasa *Sequential* este utilizată pentru a defini modele secvențiale de rețele neurale, iar *load_model* este folosită pentru a încărca un model preantrenat.

```

1 # Arduino Received commands
2 FORWARD = 'F'
3 BACKWARD = 'B'
4 LEFT = 'L'
5 RIGHT = 'R'
6 STOP = 'S'
```

Pentru a crea un program care ne oferă un context cât mai clar în timpul realizării programului, vom declara următoarele constante și variabile, care vor reprezenta comenzi pentru funcțiile de deplasare ale mașinii Arduino.

```

1 # Initialize mediapipe
2 mpHands = mp.solutions.hands
3 hands = mpHands.Hands(max_num_hands=1, min_detection_confidence=0.8)
4 mpDraw = mp.solutions.drawing_utils
5
6 # Load the gesture recognizer model
7 model = load_model('mp_hand_gesture')
8
9 # Load class names
10 with open('gesture.names', 'r') as f:
11     classNames = f.read().split('\n')
12
13 # Initialize the webcam
14 cap = cv2.VideoCapture(0)

```

Această secvență de cod se referă la inițializarea componentelor necesare pentru recunoașterea gesturilor de mână:

- **mpHands = mp.solutions.hands:** Această linie de cod atribuie variabilei *mpHands* modulul *mp.solutions.hands* din biblioteca *MediaPipe* și este responsabilă pentru recunoașterea gesturilor de mâna.
- **hands = mpHands.Hands(...):**
Această linie creează o instanță a obiectului *Hands* din modulul *mp.Hands*. Parametrul *maxnum_hands=1* specifică numărul maxim de mâini care dorim să fie detectate, iar parametrul *min_detection_confidence=0.8* specifică pragul minim de încredere dorit atunci când se detectează un gest.
- **model = load_model('mp_hand_gesture'):** Această linie încarcă modelul de recunoaștere a gesturilor de mâna din fișierul *mp_hand_gesture*, model care a fost preantrenat și va fi utilizat pentru acest proiect.
- Următoarele secvențe din sintaxă se vor ocupa de deschiderea fișierului *gesture.names* și atribuirea denumirilor gesturilor în clasa *classNames*, pentru a atribui etichete corespunzătoare predicțiilor modelului.
- **cap = cv2.VideoCapture(0):** Această linie inițializează capturarea video utilizând camera laptopului. În funcție de numărul de camere disponibile pe unitate, parametrul metodei *VideoCapture* va fi modificat pentru a determina ce camera va fi utilizată pentru înregistrare. În cazul laptopului folosit, acesta are o singură cameră, parametrul necesar va fi 0.

Următoarele secvențe de cod combină detectarea gesturilor de mâna sub formă de landmark-uri, predicția gesturilor mâinii și afișarea rezultatului pe cadrul înregistrat de camera laptopului astfel:

```
1 # Read each frame from the webcam
2 _, frame = cap.read()
3
4 x, y, c = frame.shape
5
6 # Flip the frame vertically
7 frame = cv2.flip(frame, 1)
8 framergb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
9
10 # Get hand landmark prediction
11 result = hands.process(framergb)
12
13 className = ''
```

- **cap.read()**: Această funcție citește fiecare cadru din sursa video și returnează două valori, o valoare booleană care indică dacă citirea a fost realizată sau nu, alături de cadrul citit.
- **frame.shape**: Această funcție obține dimensiunile cadrului citit și atribuie în variabilele *x*, *y* și *c* lățimea, înălțimea și, respectiv, numărul de canale de culoare ale cadrului.
- **cv2.flip()**: Această funcție inversează imaginea camerei pentru a obține o imagine reflectată. Al doilea parametru (1) indică oglindirea verticală.
- **cv2.cvtColor()**: Această funcție realizează conversia formatului de culoare a cadrului din BGR (Blue,Green,Red) în RGB (Red, Green, Blue). Librăria *OpenCV* citește imaginile în formatul BGR, dar librăria *MediaPipe* lucrează cu imagini în formatul RGB.
- **hands.process()**: Această linie procesează cadrul în format RGB pentru a detecta reperele(landmark-urile) mâinii. Un landmark în cazul unei mâini poate fi vârful unui deget, încheietura mâinii etc..

```

1 # post process the result
2     if result.multi_hand_landmarks:
3         landmarks = []
4         for handlms in result.multi_hand_landmarks:
5             for lm in handlms.landmark:
6                 lmx = int(lm.x * x)
7                 lmy = int(lm.y * y)
8
9             landmarks.append([lmx, lmy])
10
11     # Drawing landmarks on frames
12     mpDraw.draw_landmarks(frame, handlms, mpHands.HAND_CONNECTIONS
13 )
14
15     # Predict gesture
16     prediction = model.predict([landmarks])
17     classID = np.argmax(prediction)
        className = classNames[classID]

```

- **for handlms în result.multi_hand_landmarks:** Această buclă parcurge toate landmark-urile mâinii detectate din variabilă *result*.
- **lmx = int(lm.x * x) și lmy = int(lm.y * y):** Aceste instrucțiuni calculează coordonatele *x* și *y* ale fiecărui landmark, scalându-le în funcție de dimensiunea cadrului. Ulterior, valorile lui *lmx* și *lmy* sunt atribuite în lista *landmarks*.
- **mpDraw.draw_landmarks():** Această funcție desenează landmark-urile mâinii pe cadru. Parametrul *mpHands.HAND_CONNECTIONS* specifică conectarea punctelor de landmark, desenând linii între ele.
- **model.predict():** Această linie realizează predictia gestului de mâna în funcție de landmark-urile detectate.
- **np.argmax(prediction):** Această linie găsește indicele cu cea mai mare valoare din vectorul de predicție, care reprezintă clasa/gestul prezis.
- **className = classNames[classID]:** Această linie desemnează și atribuie în variabila *className* clasa/gestul prezis.

```

1 # Show the prediction on the frame
2 cv2.putText(frame, className, (10, 50), cv2.FONT_HERSHEY_SIMPLEX, 1,
3 (50, 122, 255), 2, cv2.LINE_AA)
4
5 # Show the final output
6 cv2.imshow("Output", frame)
7
8 if cv2.waitKey(1) == ord('q'):
9     break

```

- **cv2.putText()** și **cv2.imshow()**: Aceste funcții adaugă textul clasei prezise pe cadru și îl afișează.
- **cv2.waitKey(1) == ord('q')**: Aceasta este condiția de ieșire din program. Programul de detectare al gesturilor de mâna va rula până când utilizatorul apasă tasta *q*.

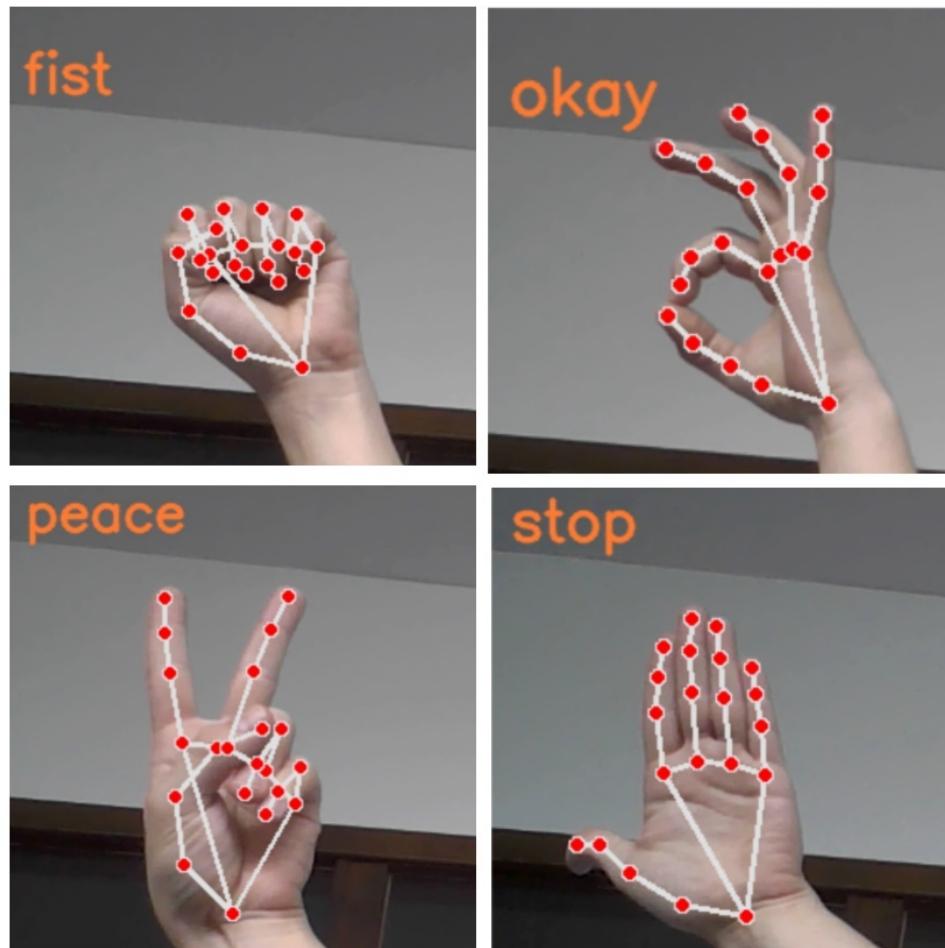


Figura 3.1: Câteva din gesturile recunoscute de program.

Astfel, acest cod implementează un sistem de recunoaștere a gesturilor mâinii în timp real, utilizat pentru a trimite comenzi de deplasare către mașină. Gestul "Okay" este asociat cu comanda de a merge înainte, gestul "Stop" este asociat cu comanda de a merge înapoi, gestul "Fist" este asociat cu comanda de a vira stânga, iar gestul "Peace" este asociat cu comanda de a vira dreapta.

Folosind MediaPipe și TensorFlow, codul detectează landmark-urile mâinii în fiecare cadru capturat de la camera laptopului. Apoi, se utilizează un model preantrenat pentru a prezice gestul mâinii pe baza landmark-urilor detectate. În funcție de gestul detectat, se trimit comenzi corespunzătoare prin portul serial către Arduino, care controlează mașina.

Rezultatul gestului detectat este afișat pe cadru și este afișat într-o fereastră separată. Codul rulează într-o buclă infinită până când utilizatorul apasă tasta 'q', moment în care camera web este eliberată și ferestrele active sunt închise. Acest cod poate fi utilizat pentru a crea o aplicație interactivă care permite controlul unei mașini Arduino folosind gesturile mâinii în timp real.

```
1/1 [=====] - 0s 56ms/step
1/1 [=====] - 0s 54ms/step
1/1 [=====] - 0s 56ms/step
1/1 [=====] - 0s 54ms/step
1/1 [=====] - 0s 56ms/step
1/1 [=====] - 0s 57ms/step
1/1 [=====] - 0s 55ms/step
1/1 [=====] - 0s 56ms/step
1/1 [=====] - 0s 56ms/step
1/1 [=====] - 0s 57ms/step
1/1 [=====] - 0s 57ms/step
1/1 [=====] - 0s 35ms/step
1/1 [=====] - 0s 36ms/step
1/1 [=====] - 0s 38ms/step
```

Figura 3.2: Datele de ieșire din consolă cu timpii de detecție al gesturilor.

Fiecare linie din output reprezintă o iterare a procesului de predicție pentru un cadru de imagine. În cazul de față, avem 15 linii, ceea ce înseamnă că se fac 15 predicții pentru 15 cadre de imagine. Fiecare linie conține informații despre progresul procesului de predicție și timpul de execuție al unei iterări. Timpul de execuție al unei iterări este afișat în milisecunde(ms) și variază între 35ms și 62ms. Aceste informații sunt utile pentru a evalua performanța și timpul de execuție al procesului de predicție a gesturilor mâinii. În cazul de față, procesul pare să fie destul de rapid, cu un timp mediu de execuție de aproximativ 51ms pe iterare.

Capitolul 4

Concluzii

Realizarea unei mașini Arduino s-a dovedit a fi o aplicație interesantă și complexă, în care am integrat cu succes trei domenii tehnologice: robotică, electronică și inteligență artificială. Această aplicație deschide noi perspective în viitor, oferind utilizatorilor posibilitatea de a controla o mașină teleghidată nu doar prin metode tradiționale, ci și prin recunoașterea gesturilor de mâna. Elementul de interactivitate adus de acest proiect este un aspect cheie, sporind experiența utilizatorului și facilitând o comunicare intuitivă cu mașina.

Folosirea tehnologiei Arduino în acest proiect se dovedește a fi foarte benefică și promițătoare. Arduino oferă o platformă flexibilă și ușor de programat, care permite integrarea cu diverse componente și senzori, facilitând dezvoltarea de aplicații complexe. În plus, combinarea tehnologiei Arduino cu inteligența artificială oferă noi oportunități și posibilități de îmbunătățire continuă a mașinii. Acest proiect are un mare potențial pentru a avea viitoare iterații și îmbunătățiri. Prin continuarea dezvoltării și optimizării algoritmilor de recunoaștere a gesturilor, se pot adăuga mai multe funcționalități și comenzi, permitând utilizatorilor să interacționeze într-un mod mai complex cu mașina.

4.1 Probleme întâmpinate și îmbunătățiri

În cadrul acestui proiect, am reușit să dezvoltăm o mașină Arduino controlată prin gesturi de mâna, depășind numeroase provocări și probleme întâmpinate pe parcurs:

- Am identificat și remediat erorile de montaj și de sintaxă, asigurându-ne că mașina se deplasează corect în funcție de comenziile primite.
- Am implementat un divizor de tensiune pentru a proteja modulul Bluetooth HC-05.
- Am ales bateriile 18650 ca soluție optimă, datorită capacității lor de stocare a energiei, costului redus și ușurinței în utilizare.

- Am adresat și rezolvat calitatea componentelor, în cazul motoarelor TT, prin prelucrarea manuală a axelor.
- Am decis utilizarea Python pentru o conexiune Bluetooth mai ușoară și mai elegantă.

Acstea eforturi ne-au permis să obținem o funcționalitate corectă și să oferim o experiență de control a mașinii Arduino prin interpretarea gesturilor de mână, dezvoltând abilități în domeniile roboticii, electronicii și inteligenței artificiale.

Pe lângă realizările actuale, există posibilități de îmbunătățire a proiectului în viitor. Printre acestea se numără crearea unei carcase modelate și printate 3D pentru a oferi un aspect mai atractiv, implementarea mai multor gesturi de mână pentru diferite comenzi de manevrare, recunoașterea facială și salvarea unui set de comenzi specifice pentru fiecare utilizator. De asemenea, ar putea fi dezvoltată o aplicație mobilă pentru recunoașterea gesturilor de mână, sporind portabilitatea sistemului.

In concluzie, prin identificarea și rezolvarea promptă a acestor probleme, am reușit să obținem o funcționalitate corectă și să oferim o experiență de control a ansamblului Arduino prin interpretarea gesturilor de mână. Dezvoltarea acestui proiect m-a ajutat să îmi dezvolt abilități în toate domeniile acoperite. În ceea ce privește robotica, am aprofundat cunoștințele și am înțeles cum se abordează un proiect de la zero, inclusiv selecția componentelor potrivite. Pe partea de electronică, am trecut de la etapa teoretică la cea practică, experimentând personal dificultățile și soluțiile pentru a remedia problemele într-un circuit electric. În plus, am reușit să aplic un model preantrenat pentru detectarea gesturilor de mână și realizat conexiunea între computer și mașina. Acest proiect m-a ajutat să înțeleg importanța integrării corecte a componentelor electronice și a dezvoltat abilitățile mele în toate aceste domenii. Aceste experiențe ne-au pregătit pentru a explora și implementa îmbunătățiri viitoare, asigurându-ne că proiectul rămâne relevant și inovator în domeniul său.

Bibliografie

- [1] Alexnld, *DIY 4WD Smart Robot Car Chassis Kits with Magneto Speed Encoder*, URL: <https://alexnld.com/product/diy-4wd-smart-robot-car-chassis-kits-with-magneto-speed-encoder-for-arduino-51/>.
- [2] Arduino, *Arduino Uno Rev3*, Disponibil la adresa, URL: <https://docs.arduino.cc/hardware/uno-rev3>.
- [3] Arduino, *Arduino Uno Rev3 Documentation*, URL: <https://docs.arduino.cc/hardware/uno-rev3>.
- [4] Chinadaier, *KCD1-101 10 Amp Rocker Switch*, URL: <https://www.chinadaier.com/kcd1-101-10-amp-rocker-switch/>.
- [5] Build Electronic Circuits, *H-Bridge - Build Electronic Circuits*, Disponibil la adresa, URL: <https://www.build-electronic-circuits.com/h-bridge/>.
- [6] Components101, *ATmega328P Pinout, Features, Datasheet & Applications*, URL: <https://components101.com/microcontrollers/atmega328p-pinout-features-datasheet>.
- [7] Components101, *HC-05 Bluetooth Module*, Iun. 2021, URL: <https://components101.com/wireless/hc-05-bluetooth-module>.
- [8] Components101, *L298N Motor Driver Module*, Apr. 2021, URL: <https://components101.com/modules/l293n-motor-driver-module>.
- [9] Digital Electronics, *L298N Motor Driver Module 2A*, URL: <https://digitalelectronics.lk/product/l298n-motor-driver-module-2a/>.
- [10] Little Bird Electronics, *Arduino Uno R3*, Disponibil la adresa, URL: <https://littlebirdelectronics.com.au/products/arduino-uno-r3>.
- [11] Paul Evans, *DC Motor Explained*, Apr. 2020, URL: <https://theengineeringmindset.com/dc-motor-explained/>.
- [12] Finglai, *KCD1-101-10-B Rocker Switch*, URL: <https://www.finglai.com/products/switches/rocker-switches/KCD-other/KCD1-101-10-B.html>.
- [13] GeeksforGeeks, *Viteza de baud și importanța sa*, Disponibil la adresa, URL: <https://www.geeksforgeeks.org/baud-rate-and-its-importance/>.

- [14] Ian Goodfellow, Yoshua Bengio și Aaron Courville, *Deep Learning*, MIT Press, 2016, URL: <http://www.deeplearningbook.org>.
- [15] Adafruit Industries, *Feather 32u4 Bluefruit LE*, Disponibil la adresa, URL: <https://www.adafruit.com/product/3777>.
- [16] Battery Junction, *Panasonic NCR18650B 3400mAh Battery*, URL: <https://www.batteryjunction.com/panasonic-nr18650b-3400.html>.
- [17] Keras, *About Keras*, Disponibilă la adresa, URL: <https://keras.io/about/>.
- [18] Fenix Lighting, *The ultimate guide to the 18650 battery*, Disponibil la adresa, URL: <https://www.fenixlighting.com/blogs/news/the-ultimate-guide-to-the-18650-battery>.
- [19] Nicholas Brown, *Introduction To PWM: How Pulse Width Modulation Works*, URL: <https://www.kompulsa.com/introduction-pwm-pulse-width-modulation-works/>.
- [20] OpenCV, *About OpenCV*, Disponibil la adresa, URL: <https://opencv.org/about/>.
- [21] Steve Oualline, *Practical C++ Programming*, 2nd, O'Reilly Media, 2003, ISBN: 0-596-00419-2.
- [22] Zachariah Peterson, *What is a PCB?*, Oct. 2020, URL: <https://resources.altium.com/p/what-is-a-pcb>.
- [23] Robot Edu, *Bracket and Holder for 18650 Battery*, URL: <https://www.admin.robotedu.my/bracket-and-holder/18650>.
- [24] SkyGenius, *Beginner's Guide: 18650 Battery*, Iul. 2021, URL: <https://skygenius.cc/blogs/guides/beginners-guide-18650-battery>.
- [25] Django Stars, *Blog Django Stars*, Disponibil la adresa, URL: <https://djangostars.com/blog/>.
- [26] Google Play Store, *Bluetooth RC Controller - Apps on Google Play*, Disponibil la adresa, URL: <https://play.google.com/store/apps/details?id=braulio.calle.bluetoothRCcontroller&pli=1>.
- [27] TensorFlow, *Learn TensorFlow*, Disponibil la adresa, URL: <https://www.tensorflow.org/learn>.
- [28] Total Phase, *Understanding Differences Between UART and USB*, Ian. 2022, URL: <https://www.totalphase.com/blog/2022/01/understanding-differences-between-uart-and-usb/>.
- [29] Wikipedia, *Fritzing*, URL: <https://en.wikipedia.org/wiki/Fritzing>.
- [30] Wikipedia, *Lithium Polymer Battery*, URL: https://en.wikipedia.org/wiki/Lithium_polymer_battery.

- [31] Wikipedia, *Lithium-ion battery*, URL: https://en.wikipedia.org/wiki/Lithium-ion_battery.
- [32] Wikipedia, *Voltage divider*, URL: https://en.wikipedia.org/wiki/Voltage_divider.
- [33] Kris Wouk, *What Is mAh (Milliamp Hours)?*, Jun. 2022, URL: <https://www.howtogeek.com/804692/what-is-mah/>.