UNIVERSITATEA DIN
BUCUREȘTI

FACULTATEA DE
MATEMATICĂ ȘI
INFORMATICĂ

SPECIALIZAREA
TEHNOLOGIA INFORMAȚIEI

Lucrare de licență

MAȘINĂ SEMIAUTONOMĂ CU ASISTENT DE CONDUCERE

Absolvent
Enescu Horia Teodor Gabriel

Coordonatori științifici
Conf. Dr. Alexe Bogdan
Drd. Dumitriu Andrei

București, iunie 2025

Rezumat

Acest proiect prezintă dezvoltarea unui prototip de vehicul semiautonom cu tracțiune integrală, dotat cu un asistent de conducere activ. Mașina este concepută astfel încât să simuleze comportamentul dinamic al unui vehicul real, utilizând un mecanism de direcție de tip Ackermann, similar cu cel folosit la autoturismele moderne. Aceasta este controlată în mod direct de către un utilizator prin intermediul unei telecomenzi, însă sistemul este capabil să intervină autonom în situații critice, având scopul de a preveni accidentele și de a crește siguranța rutieră.

Prototipul integrează mai multe tehnologii avansate din domeniul roboticii și al vederii artificiale, dintre care se remarcă funcționalitatea de detecție automată a semnelor de circulație. Prin combinarea controlului manual cu funcții autonome inteligente, proiectul explorează posibilitatea de tranziție către vehicule complet autonome.

Abstract

This project presents the development of a semi-autonomous all-wheel-drive vehicle prototype equipped with an active driving assistant. The car is designed to simulate the dynamic behavior of a real vehicle, using an Ackermann steering mechanism, similar to those found in modern automobiles. It is directly controlled by a user via remote control; however, the system is capable of autonomous intervention in critical situations, aiming to prevent accidents and increase road safety.

The prototype integrates several advanced technologies from the fields of robotics and computer vision, among which the automatic traffic sign detection functionality stands out. By combining manual control with intelligent autonomous features, the project explores the potential transition toward fully autonomous vehicles.

Cuprins

Listă de figuri

Capitolul 1

Introducere

În ultimii ani, domeniul conducerii autonome a evoluat semnificativ și a devenit prioritar pentru potențialul său uriaș de a reduce drastic numărul accidentelor rutiere, de a crește siguranța participanților la trafic și de a fluidiza circulația rutieră. În prezent, accidentele rutiere reprezintă o problemă gravă la nivel global, iar marea majoritate a acestora sunt cauzate de erori umane precum lipsa de atenție, oboseala sau nerespectarea semnificației semnelor de circulație. Astfel, introducerea unor sisteme avansate de asistență pentru șoferi este vitală pentru îmbunătățirea siguranței pe șosele.

Vehiculele semiautonome oferă o soluție intermediară între mașinile tradiționale, conduse integral de operatori umani, și vehiculele complet autonome. Acestea permit interacțiunea continuă între șofer și sistemele automate inteligente, îmbinând avantajele percepției umane cu precizia și viteza de reacție specifice asistentului de conducere.

Proiectul de față propune dezvoltarea unui model de mașină semiautonomă, controlată printr-o telecomandă de către un operator uman, care beneficiază de un sistem de asistență rutieră. Sistemul creat are rolul de a detecta și interpreta semnele de circulație și marcajele benzilor rutiere, intervenind activ atunci când șoferul uman ignoră semnificația semnelor sau părăsește accidental banda de circulație. Prin aceasta, se urmărește crearea unei experiențe mai sigure în trafic și prevenirea potențialelor accidente.

Problema principală abordată în această lucrare este reducerea erorilor umane în timpul condusului cauzate de oboseală, lipsă de atenție sau distragere, întrucât majoritatea accidentelor rutiere ar putea fi evitate dacă șoferii sunt avertizați sau ajutați să reacționeze prompt în fața unor situații neprevăzute. Astfel, este evidentă necesitatea implementării unor sisteme inteligente care să monitorizeze permanent traficul și să sprijine deciziile șoferului.

Soluția propusă în această lucrare constă într-o mașină dotată cu senzori de detecție a benzii și o cameră video conectată la un sistem inteligent capabil să proceseze imaginile preluate în timp real. Sistemul analizează continuu drumul și identifică semnele rutiere. Atunci când este detectat un semn nou, operatorul este notificat imediat. În cazul în care operatorul nu respectă indicațiile semnului (spre exemplu, nu oprește la STOP)

sau părăsește banda involuntar, sistemul intervine automat pentru a evita un accident iminent.

Această lucrare este structurată în următoarele capitole principale:

În capitolul al doilea se prezintă noțiunile teoretice esențiale pentru înțelegerea tehnologiilor folosite în cadrul proiectului. Capitolul al treilea se concentrează pe realizarea și integrarea componentelor fizice ale sistemului, cum ar fi construcția vehiculului, a telecomenzii și a traseului. Arhitectura software, algoritmii utilizați pentru procesarea în timp real și dezvoltarea logicii sistemului sunt prezentate în al patrulea capitol. Concluziile și observațiile la finalul realizării proiectului, evidențierea rezultatelor obținute și perspectivele pentru dezvoltări viitoare sunt prezentate în ultimul capitol.

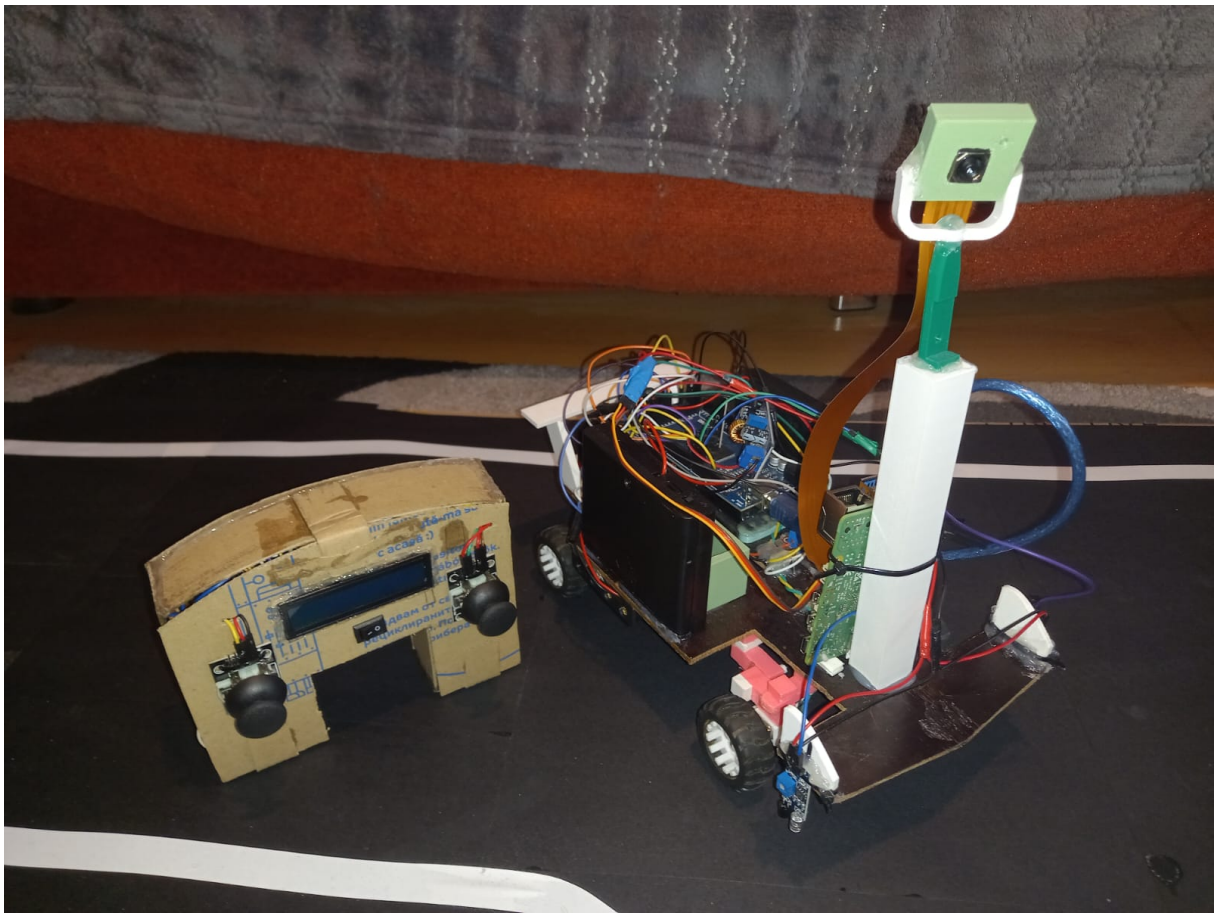


Figura 1.1: Vehiculul și telecomanda în forma finală

Capitolul 2

Preliminarii

2.1 Sistemul Ackermann

Sistemul de direcție Ackermann reprezintă un mecanism utilizat la vehiculele pe 4 roți care asigură o manevrabilitate optimă în viraje.

Propusă inițial de către Georg Lankensperger în anul 1816 și patentată ulterior de inginerul Rudolf Ackermann în 1818 [1], principiul de funcționare al geometriei Ackermann stă la baza majorității vehiculelor moderne.

În momentul efectuării virajelor, fiecare roată din față a mașinii ar trebui să urmeze un arc de cerc distinct, însă ambele arce trebuie să fie concentrice, astfel încât să aibă același centru de rotație comun.

Presupunem:

- L = distanța dintre axele față și spate (ampatamentul)
- W = lățimea vehiculului la roțile din față (ecartamentul)
- R = raza de virare măsurată de la centrul vehiculului
- θ_i = unghiul de virare al roții interioare
- θ_o = unghiul de virare al roții exterioare

Există două relații care descriu geometria direcției Ackermann:

$$\tan(\theta_i) = \frac{L}{R - \frac{W}{2}} \qquad (2.1) \qquad\qquad \tan(\theta_o) = \frac{L}{R + \frac{W}{2}} \qquad (2.2)$$

Se observă că:

- Roata interioară virează mai mult ($\theta_i > \theta_o$)
- Liniile de direcție ale roților se întâlnesc într-un punct comun

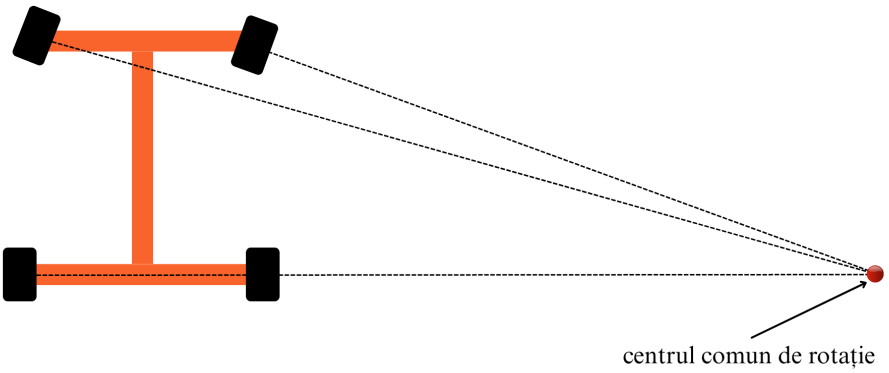


Figura 2.1: Geometria direcției Ackermann: se observă modul în care roțile sunt orientate astfel încât liniile lor de direcție să se intersecteze într-un punct comun, numit centrul de rotație.

2.2 Sistemele ADAS

ADAS[2] reprezintă un ansamblu de soluții tehnologice integrate în arhitectura electronică și software a vehiculului. Aceste sisteme nu substituie complet șoferul, ci oferă suport activ în procesul de conducere, fiind considerate etape intermediare importante în evoluția către vehiculele complet autonome.

Având o funcționare bazată pe o rețea complexă de senzori (camere video, radar, lidar și senzori ultrasonici) care trimit date către unitățile de control capabile să analizeze rapid situațiile din trafic, ADAS contribuie semnificativ la creșterea siguranței rutiere și reducerea erorilor umane care cauzează, conform unui studiu realizat de Administrația Națională pentru Siguranța Traficului Rutier din Statele Unite [3], aproximativ 94% din accidentele rutiere. Aceste tehnologii includ asistență la menținerea benzii de circulație, avertizare la coliziune frontală, control adaptiv al vitezei și detectarea pietonilor. Acest ansamblu de facilități oferă șoferului o experiență mai sigură și mai confortabilă în timpul condusului.

În practică, sistemele ADAS permit comunicarea între vehicule și infrastructura rutieră și facilitează fluidizarea traficului. Totodată, aceste tehnologii promovează un stil de conducere mai responsabil și mai ecologic, contribuind la protecția mediului.

Capitolul 3

Hardware

Un aspect definitoriu al acestui proiect îl reprezintă realizarea completă de la zero a întregului sistem, fără utilizarea componentelor preasamblate. Mașina a fost construită manual, piesă cu piesă, folosind componente individuale. Telecomanda a fost proiectată și realizată în întregime și adaptată la cerințele vehiculului. De asemenea, traseul pe care se deplasează mașina a fost conceput și construit manual, iar semnele de circulație au fost create din materiale de bază, respectând proporțiile și aspectul celor reale. Întreg ansamblul reflectă o muncă complet originală, cu accent pe construcție practică.

3.1 Mașina

3.1.1 Proiectarea și execuția șasiului

Un prim pas pentru realizarea mașinii a fost construcția șasiului. Acesta a fost realizat manual, pornind de la o placă de textolit. Materialul a fost ales datorită rigidității sale, a greutății reduse și a ușurinței în prelucrare. Dimensiunile șasiului au fost stabilite prin raportare la cotele unei mașini reale[4], în vederea obținerii unui echilibru realist între lungime și lățime.

După stabilirea cotelor, s-a decis ca șasiul să aibă o lungime de 30 de centimetri și o lățime de 15 centimetri, dimensiuni considerate optime pentru stabilitate și manevrabilitate în cadrul traseului proiectat. Placa de textolit a fost apoi marcată conform acestor dimensiuni, iar conturul șasiului a fost decupat folosind un polizor unghiular (flex). A urmat etapa de finisare, în care a fost folosită o bucată de șmirghel pentru netezirea colțurilor și a marginilor proaspăt tăiate. Rezultatul este vizibil în Figura 3.2a.

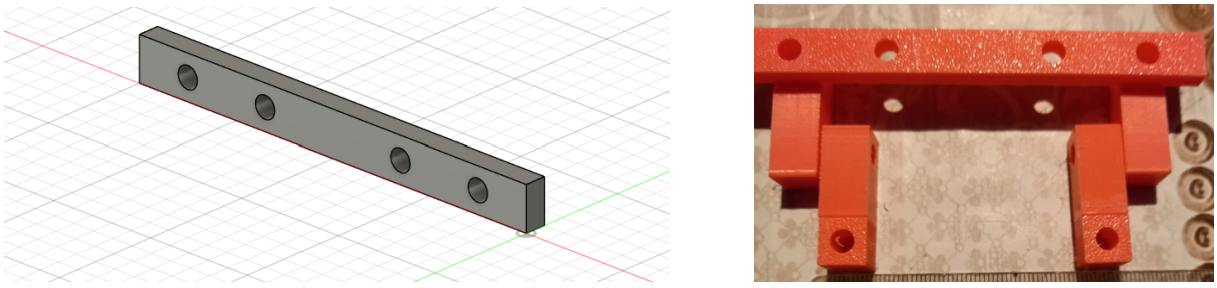
3.1.2 Mecanismul Ackermann

Pentru proiectarea pieselor specifice sistemului Ackermann, dar și a altor piese ce țin de aspect, software-ul Autodesk Fusion 360 a fost alegerea potrivită. Această aplicație a fost aleasă datorită capabilităților sale avansate de modelare 3D, dar și pentru interfața și funcțiile asemănătoare cu AutoCAD.

Componentele sistemului de direcție Ackermann au fost proiectate de la zero, măsurile lor raportându-se la dimensiunile șasiului, astfel încât să se potrivească perfect între ele și să reproducă cât mai realist principiile de funcționare ale mecanismului Ackermann.

După finalizarea proiectării pieselor în Fusion, componentele au fost exportate în format STL și pregătite pentru imprimare folosind software-ul PrusaSlicer.

Imprimarea s-a realizat pe o imprimantă Original Prusa i3 MK3S+. A fost utilizat un strat de 0,15mm. S-a optat pentru un grad de umplere (infill) de 15%, considerat suficient pentru piesele utilizate în acest proiect. Acest procent oferă un echilibru între rezistența pieselor și timpul de producție. Pe parcursul procesului, au apărut și câteva dificultăți tehnice: unele piese, având forme neregulate, se deformau la imprimare. Această problemă a fost rezolvată prin aplicarea unor setări suplimentare în PrusaSlicer, precum existența unor suporți de susținere (supports). După imprimare, fiecare piesă a fost verificată dimensional, pentru asigurarea compatibilității și îmbinării corecte cu celelalte componente. În unele cazuri, s-au efectuat mici ajustări manuale cu un cutter, apoi cu un șmirghel fin pentru netezirea suprafeței. De asemenea, bavurile care ar fi putut afecta fixarea pieselor au fost îndepărtate cu grijă. La final, componentele 3D rezultate au fost lipite folosind un lipici rezistent cu uscare rapidă, iar pentru o fixare stabilă și ușor demontabilă a întregului sistem pe șasiu, s-au utilizat 2 șuruburi și 2 piulițe.

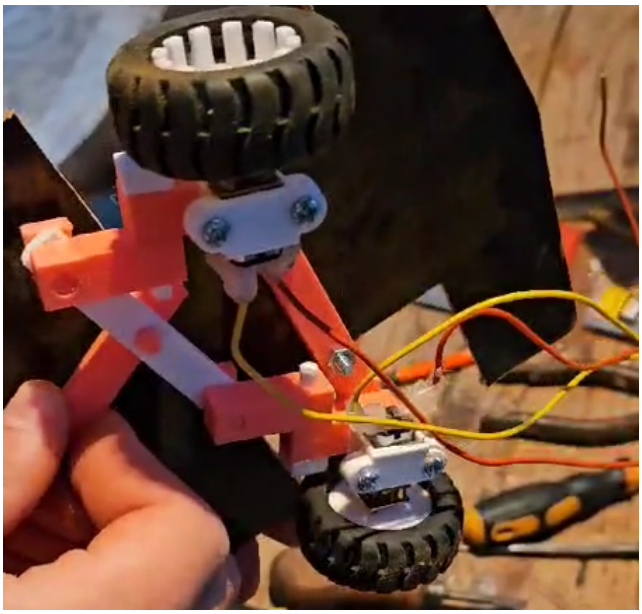


(a) Piesă proiectată în Fusion

(b) Piese printate 3D

Figura 3.1: Etapele realizării pieselor 3D: În a) se observă proiectarea unei piese, iar în b) aceeași piesă, împreună cu alte componente asemănătoare după finalizarea printării 3D

(a) Șasiul finalizat



(b) Mecanismul Ackermann

Figura 3.2: În a) se observă șasiul decupat și finalizat, realizat din placă de textolit, reprezentând baza vehiculului, iar în b) mecanismul Ackermann montat pe șasiu, evidențiind îmbinarea precisă dintre componentele printate și șasiu.

3.1.3 Integrarea componentelor

Proiectarea Hardware a mașinii a avut în vedere atât controlul în timp real al mișcării și direcției, cât și prelucrarea informațiilor provenite de la cameră. S-a acordat o atenție deosebită distribuirii uniforme a greutății pe șasiu, astfel încât să se asigure o funcționare stabilă și eficientă. Poziționarea componentelor nu a fost aleatorie, ci planificată astfel încât să se evite pe cât posibil dezechilibrele care ar putea afecta manevrabilitatea și eficiența tracțiunii.

Pentru gestionarea comunicărilor seriale și a procesării video, s-a folosit inițial un Raspberry Pi 4 cu 4 GB memorie RAM. Totuși, acest model s-a dovedit ineficient din punct de vedere al timpului de prelucrare, în special în contextul rulării în timp real. Ca urmare, a fost ales un Raspberry Pi 5 cu 8 GB memorie RAM, care oferă performanțe superioare. La acest calculator a fost conectată o cameră video Modul 3, capabilă să transmită imagini în timp real. Astfel, Raspberry Pi îndeplinește rolul de unitate centrală de procesare, prin transmiterea comenzilor către Arduino Mega prin cablul USB A-B, cât și prin gestionarea fluxului video provenit de la camera frontală.

Alimentarea calculatorului Raspberry Pi se face printr-un cablu USB Tip C, conectat la o baterie externă, deoarece alimentarea sa este una stabilă. Bateriile externe mobile sunt proiectate exact pentru astfel de cerințe și oferă protecții integrate împotriva supratensiunii sau a descărcării complete. În plus, utilizarea ei elimină necesitatea unor componente suplimentare de reglare a tensiunii.

Pentru controlul tuturor componentelor mașinii, plăcuța Arduino Mega 2560 a fost

alegerea potrivită. S-a optat pentru această plăcuță în detrimentul altor microcontrolere pentru numărul său mare de pini analogici și digitali disponibili, esențiali pentru conectarea simultană a unui număr ridicat de alte componente. De asemenea, Arduino Mega 2560 dispune de un microcontroler ATmega2560, care oferă o performanță suficientă pentru gestionarea eficientă a comenzilor primite în timp real atât de la telecomandă, cât și de la Raspberry Pi.

Tracțiunea integrală a vehiculului s-a realizat cu 4 micro motoare cu reducție, fiecare având o tensiune maximă de alimentare de 6V și 40 de rotații pe minut (rpm). S-a optat pentru aceste specificații deoarece mașina este grea și motoarele trebuie să aibă cuplu mai mare pentru a o putea propulsa. Spre deosebire de motoarele cu turație mare, care oferă viteză în detrimentul forței de tracțiune, motoarele cu 40 rpm asigură cuplu ridicat la turație joasă, având așadar capacitatea de a porni de pe loc sub sarcină. Pentru montarea lor s-au folosit 4 suporți printați 3D, care ulterior au fost lipiți pe sistemul Ackermann și pe șasiu.

Controlul motoarelor cu reducție s-a realizat cu ajutorul a două module L298N, câte unul pentru fiecare punte de acționare. Acest modul este un driver de motoare cu punte H dublă, care permite controlul motoarelor de curent continuu (DC) în ambele sensuri (înainte și înapoi). Acest tip de driver are o tensiune de operare de până la 40 de volți, posibilitatea alimentării de la o sursă externă, protecție la supracurent și fiabilitate în ceea ce privește controlul a două motoare simultan. Pentru a asigura funcționarea optimă a modulului L298N, s-a utilizat atât alimentarea de 5V de la Arduino, cât și o sursă externă formată din 4 baterii AA conectate în serie, oferind un total de 6V.

Controlul mecanismul Ackermann s-a realizat cu un servomotor MG996R având gradul de rotație 180. Acesta s-a legat la Arduino Mega pentru primirea și executarea comenzilor în timp real. Având o tensiune de operare cuprinsă între 4,8 și 7,2 volți, alimentarea servomotorului s-a făcut cu 6 baterii reîncărcabile AA de 1,2 volți fiecare, legate în serie, însumând 7,2 volți. Din motive de siguranță, s-a optat pentru un modul de coborâre a tensiunii, reglat la 7V, astfel încât să nu se depășească niciodată limita superioră de 7,2 volți recomandată de producător.

Pentru deplasarea mașinii s-au utilizat 4 roți din cauciuc, fiecare având un diametru de 42 milimetri și o lățime de 19 milimetri. Alegerea acestora s-a bazat atât pe compatibilitatea cu motoarele DC folosite, cât și pe rezistența lor la solicitări mecanice. Roțile asigură o tracțiune corespunzătoare, manevrabilitate ușoară și integrare eficientă cu ansamblul constructiv al vehiculului.

Identificarea abaterii de pe traseu s-a realizat prin intermediul a doi senzori cu infraroșu montați în partea din față a șasiului. Acești senzori funcționează prin emiterea și recepționarea unei raze infraroșii, iar diferența de reflexie a luminii permite distingerea contrastului dintre suprafața benzii și fundalul închis.

În cadrul arhitecturii hardware, s-au folosit două module NRF24L01 pentru comu-

nicarea wireless între telecomandă și mașină. Unul dintre module este configurat ca transmițător și trimite date către telecomandă, iar celălalt funcționează ca receptor, primind date de la telecomandă. Ambele sunt conectate fizic la plăcuța Arduino Mega, însă modulul receptor transmite instrucțiuni către microcontroler, gestionând astfel pornirea și direcția vehiculului.

3.2 Telecomanda

3.2.1 Carcasa

În vederea protejării componentelor și a obținerii unui aspect vizual plăcut, a fost construită o carcasă din carton pentru telecomandă. Cartonul a fost ales datorită ușurinței în prelucrare, a greutății reduse și rigidității suficiente pentru a susține greutatea elementelor interne. Componentele carcasei au fost asamblate cu un lipici rezistent și silicon.

Carcasa include balamale artificiale realizate tot din carton, precum și o piedică, oferind posibilitatea de a deschide ușor telecomanda pentru acces la interior. Închiderea capacului frontal se face cu aceeași ușurință. Totodată, aspectul vizual a fost îmbunătățit prin vopsirea manuală a întregii carcase.

3.2.2 Integrarea componentelor

Gestionarea logicii sistemului a fost realizată cu ajutorul unei plăcuțe Arduino Uno, aleasă pentru simplitatea în utilizare, a dimensiunilor compacte și a numărului adecvat de pini analogici și digitali necesari acestei etape. Alimentarea acesteia s-a făcut prin intermediul unui suport de 4 baterii AA conectate în serie, furnizând o tensiune de aproximativ 6V, suficientă pentru a asigura o funcționare stabilă a plăcii.

Controlul direcției vehiculului a fost realizat cu ajutorul a două joystickuri analogice, fiecare prevăzut cu două axe de mișcare. În cadrul acestui proiect a fost utilizată o singură axă de la fiecare joystick. Unul dintre joystickuri este responsabil pentru direcționare, în timp ce al doilea controlează deplasarea înainte-înapoi.

Pentru a putea informa operatorul uman, a fost montat un ecran LCD pe carcasa telecomenzii. Prin intermediul acestuia, se poate schimba modul de circulație și se afișează date din trafic în timp real.

La fel ca în cazul mașinii, telecomandă necesită două module NRF24L01 pentru comunicarea wireless. La fiecare iterație a programului, valorile colectate de la joystickuri sunt trimise prin modulul transmițător către vehicul. În același timp, al doilea modul, configurat ca receptor, primește date de la mașină, precum semnele de circulație care urmează sau informații privind intervenția asistentului de conducere, care sunt ulterior afișate pe ecranul LCD.

3.3 Comunicarea între componente

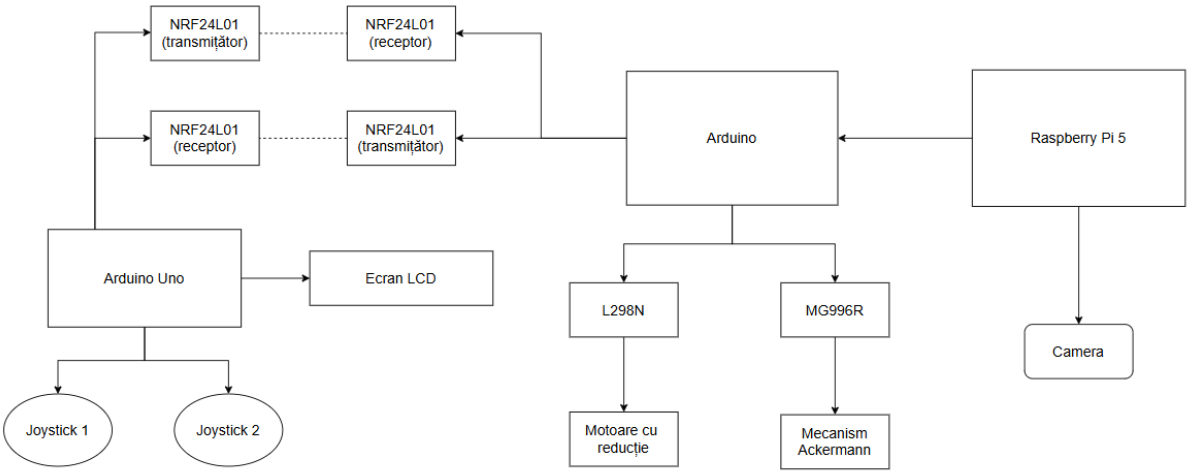


Figura 3.3: Schema comunicării între telecomandă și mașină

Comunicarea radio între modulele NRF24L01 reprezintă o componentă importantă în cadrul proiectului pentru controlul wireless. Aceste dispozitive operează în bandă de 2,4 GHz, și sunt capabile să transmită și să recepționeze date cu viteze de până la 2 Mbps. Comunicarea se realizează pe principiul master-slave (transmițător-receptor), unde un modul acționează ca transmițător (TX), iar celălalt ca receptor (RX). Cu un consum redus de energie, modulul NRF24L01 are o rază de acțiune cuprinsă între 10 și 20 de metri în spațiile închise, și până la 100 de metri în exterior.

3.4 Traseul și semnele de circulație

Pentru testarea funcționalităților vehiculului ghidat prin telecomandă, a fost realizat un traseu special, conceput să reproducă în miniatură condiții cât mai apropiate de cele întâlnite în mediul rutier real. Scopul principal al acestui traseu este de a oferi un cadru controlat în care vehiculul să poată fi testat în diverse scenarii de deplasare, orientare și interacțiune cu semnele de circulație.

Baza traseului a fost construită din carton, ușor de tăiat și suficient de stabil pentru a susține deplasarea vehiculului. Suprafața carosabilă a fost realizată prin lipirea unor foi negre, care au redat cât mai asemănător textura și culoarea asfaltului. Peste această suprafață au fost trasate marcajele rutiere cu bandă izolatoare albă, aleasă pentru contrastul clar. Marcajele includ delimitarea unei benzi de circulație și reprezentarea unei treceri de pietoni, folosind linii paralele și spațiere regulată.

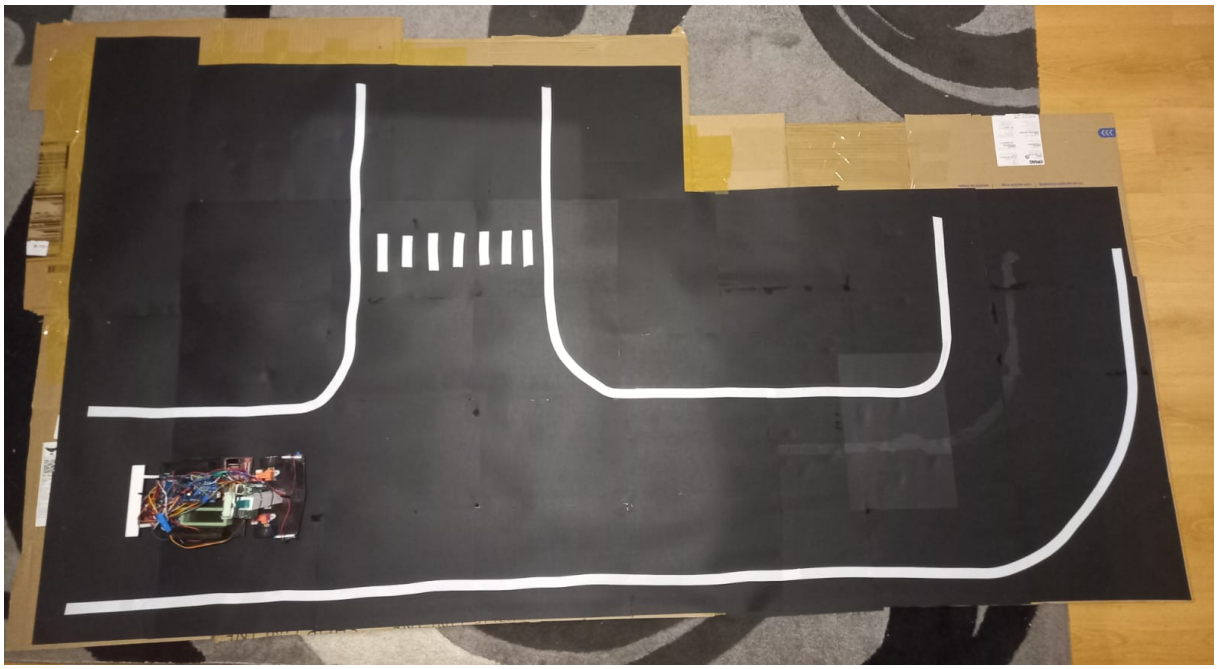
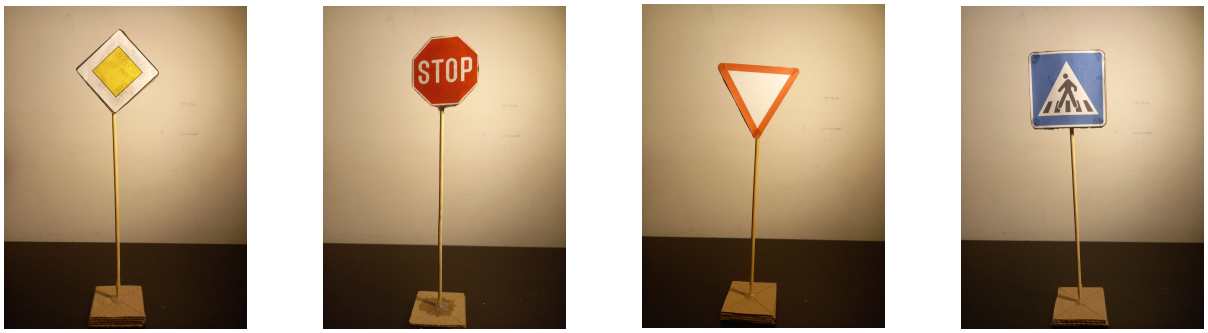


Figura 3.4: Traseul mașinii văzut de sus; se observă trecerea de pietoni și marcajul ce delimitează banda de circulație

Pe lângă traseu, au fost realizate 4 semne diferite de circulație: STOP, PRIORITATE, CEDEAZĂ TRECEREA, TRECERE DE PIETONI. Fiecare semn este compus din trei părți: panoul informativ, suportul vertical și baza.

Panoul cu semnul propriu-zis a fost decupat din carton în forme diferite (hexagon pentru STOP, triunghi pentru CEDEAZĂ TRECEREA și pătrat cu vârfurile rotunjite pentru PRIORITATE și TRECERE DE PIETONI), peste care au fost lipite imagini reprezentând semne reale, printate la scară redusă. Pentru suportul vertical s-au utilizat bețe subțiri de lemn, iar pentru bază s-au folosit pătrate din carton, menite să mențină semnele în poziția verticală. S-a optat pentru aceste materiale pentru că sunt accesibile și ușor de prelucrat astfel încât semnul de circulație rezultat să fie realist.



(a) PRIORITATE (b) STOP (c) CEDEAZĂ (d) TRECERE

Figura 3.5: Semnele de circulație rezultate; se observă panoul cu semnul, suportul vertical din lemn și baza din carton

Capitolul 4

Software

4.1 Sistemul de operare instalat pe Raspberry Pi 5

Pentru a putea implementa și testa componentele software ale proiectului, a fost necesară configurarea mediului de lucru pe dispozitivul Raspberry Pi 5. Sistemul de operare instalat este Raspberry Pi OS (64-bit), o distribuție oficială bazată pe Linux.

Raspberry Pi OS vine în 2 variante: versiunea pe 64 de biți și versiunea pe 32 de biți. În cadrul proiectului a fost aleasă cea pe 64 de biți.

Pentru instalarea sistemului de operare pe dispozitiv a fost utilizat programul oficial Raspberry Pi Imager, o unealtă grafică disponibilă gratuit pe site-ul oficial al Raspberry Pi[5]. Aplicația a permis instalarea ușoară a sistemului de operare pe cardul microSD, care ulterior a fost introdus în slotul dedicat plăcii. Având totodată o interfață intuitivă și ușor de înțeles, s-au configurat diverse setări, precum numele și parola dispozitivului, iar în final s-a realizat conectarea la o rețea Wi-Fi.

Procesul de instalare a constat în următorii pași:

- Descărcarea și instalarea aplicației Raspberry Pi Imager pe calculatorul personal
- Alegerea sistemului de operare din lista pusă la dispoziție de aplicație
- Selectarea cardului microSD
- Configurarea setărilor opționale și obligatorii (credențiale, conexiune wi-fi)
- Introducerea cardului microSD în Raspberry Pi 5 și pornirea sistemului

După instalarea sistemului de operare, au fost rulate în terminal comenzile *sudo apt update* și *sudo apt upgrade*. Prima comandă are rolul de a sincroniza lista locală de pachete cu cele mai recente versiuni disponibile, pe când a doua comandă instalează efectiv pachetele actualizate și remediază posibile vulnerabilități. Pentru a controla de la distanță interfața Raspberry Pi, a fost utilizată aplicația TigerVNC, iar pentru a scrie cod, s-a folosit editorul Visual Studio Code[6].

4.2 Algoritmul YOLOv8

YOLOv8[7] este un algoritm lansat de compania Ultralytics[8] și este folosit preponderent pentru detecția de obiecte în imagini. Modelul YOLOv8 este capabil să identifice simultan poziția prin coordonate, clasa obiectului și scorul de încredere asociat fiecărei detecții. Pentru antrenarea modelului este necesar un set de date etichetat, care trebuie împărțit în mod standard în trei subseturi:

- Imagini de antrenare - sunt folosite pentru ajustarea parametrilor. Aceste imagini conțin obiectele de interes etichetate corespunzător, iar modelul învață să le recunoască.
- Imagini de validare - sunt utilizate pe parcursul antrenării pentru a evalua performanța temporară a modelului, fără ca acestea să influențeze direct procesul de învățare.
- Imagini de testare - sunt complet separate de procesul de antrenare și validare, utilizarea lor fiind la finalul antrenării, pentru a obține o evaluare obiectivă a performanței modelului pe date noi, nevăzute.

Pentru formarea setului de date destinat antrenării modelului a fost creat un proiect nou pe platforma online Roboflow[9], un instrument care facilitează gestionarea imaginilor și a adnotărilor necesare pentru antrenarea algoritmilor de inteligență artificială.

Adnotarea imaginilor este un proces esențial în antrenarea algoritmilor de tip YOLOv8 și constă în marcarea cât mai precisă a obiectelor de interes din imagini prin trasarea unor dreptunghiuri numite bounding boxes ce încadrează zona dorită. Aceste dreptunghiuri sunt însoțite de etichete corespunzătoare clasei din care obiectul face parte. Prin acest proces, algoritmul YOLOv8 învață să recunoască obiecte specifice, să identifice poziția exactă și să clasifice obiectele detectate corect în timpul utilizării reale.

Pentru acest proiect, procesul de adnotare s-a realizat integral în Roboflow. Având o interfață intuitivă și funcționalități avansate, a permis o adnotare rapidă, precisă și eficientă. Motivul principal pentru care adnotarea a fost necesară în cadrul acestui proiect e reprezentat de nevoia unui set de date cât mai divers, cu accent pe imaginile realizate de camera montată pe vehicul.

Înainte de a putea adnota imagini, este necesară definirea claselor. În lucrarea de față, s-au definit patru clase distincte, având numele semnelor de circulație care trebuie detectate: STOP, CEDEAZĂ TRECEREA, PRIORITATE, TRECERE DE PIETONI.

După definirea claselor, următoarea etapă constă în formarea setului de imagini. În general, cu cât setul este mai mare și mai divers, cu atât și performanța modelului este mai ridicată. În cadrul acestui proiect, setul conține 2628 de imagini, dintre care aproximativ jumătate au fost preluate de pe internet, iar cealaltă jumătate realizate cu ajutorul unui telefon mobil, dar și prin intermediul camerei montate pe vehicul. Această abordare mixtă, cu imagini de pe internet și imagini personale, a fost esențială deoarece imaginile preluate de pe internet au contribuit la o diversitate sporită în ceea ce privește mediile

și perspectivele, în timp ce imaginile personale reflectă condițiile specifice ale vehiculului utilizat în proiect.

Setul total de 2628 de imagini a fost împărțit astfel: 1831 de imagini au fost alocate pentru antrenare (aproximativ 70%), 727 pentru validare (aproximativ 28%) și 70 pentru testare (aproximativ 2%), această împărțire urmărind să asigure o antrenare eficientă și o evaluare corectă a performanței modelului. Un aspect important este reprezentat de unicitatea setului de date, întrucât toate imaginile au fost adnotate exclusiv pentru acest proiect, nefiind preluate imagini pre-adnotate din alte proiecte sau seturi de date.

După finalizarea procesului de adnotare și împărțire a imaginilor pe categorii (antrenare, validare, testare), a fost creată o nouă versiune a proiectului în Roboflow, iar setul de imagini a fost descărcat împreună cu etichetele asociate. Folderul descărcat include imaginile originale și un folder separat numit **labels**, care conține fișiere text aferente fiecărei imagini în parte. În fiecare fișier text se regăsesc informațiile despre obiectele adnotate, respectiv clasa obiectului și coordonatele dreptunghiurilor ce marchează zona de interes, exprimate în format YOLO (coordonatele x,y ale centrului dreptunghiului și lățimea și înălțimea acestuia, toate normalizate între 0 și 1).

Fișierul **data.yaml** rezultat după descărcare are un rol important, întrucât conține numărul și numele claselor, precum și căile de acces pentru fișierele cu imagini de antrenare, validare și testare. Acesta a fost verificat și modificat, pentru a asigura definirea corectă a căilor către foldere.

Figura 4.1: Exemplu de imagine adnotată din setul de date de antrenare cu semnul STOP. Se observă chenarul ce marchează zona de interes

Pentru etapa de antrenare, s-a creat un fișier Python pentru antrenare. A fost folosită funcția **train()** din biblioteca Ultralytics, cu parametrii aleși pentru a optimiza procesul și a utiliza eficient resursele disponibile. Astfel, antrenarea s-a realizat pe placa video, utilizând două procese simultane. În contextul antrenării algoritmilor de tip YOLO, parametrul numit **batch** semnifică numărul de imagini procesate simultan într-o singură etapă de antrenare înainte de actualizarea parametrilor modelului. În cadrul acestui proiect, dimensiunea lotului a fost setată la 85%, pentru a preveni riscul epuizării memoriei grafice. Imaginile au fost redimensionate la 640 de pixeli, conform recomandărilor standard YOLOv8, pentru un echilibru mai bun între performanță și precizie. De asemenea, antrenarea a fost programată să se desfășoare pe maxim 300 de epoci, dar a fost inclus și un parametru de oprire timpurie (patience), care întrerupe automat procesul dacă performanța nu se mai îmbunătățește semnificativ timp de 25 de epoci consecutive, evitând astfel supraantrenarea modelului.

La finalul etapei de antrenare, rezultatul principal este fișierul **best.pt**, care reprezintă modelul YOLOv8 antrenat complet și optimizat. Pentru evaluarea performanței modelului obținut, a fost creat un alt fișier Python care folosește **best.pt** și permite testarea modelului pe imaginile realizate cu ajutorul camerei montate pe vehicul, salvate în folderul imaginilor de testare. Funcția utilizată, **predict()**, analizează imaginile și salvează automat rezultatele obținute într-un folder separat, pentru o analiză ulterioară.

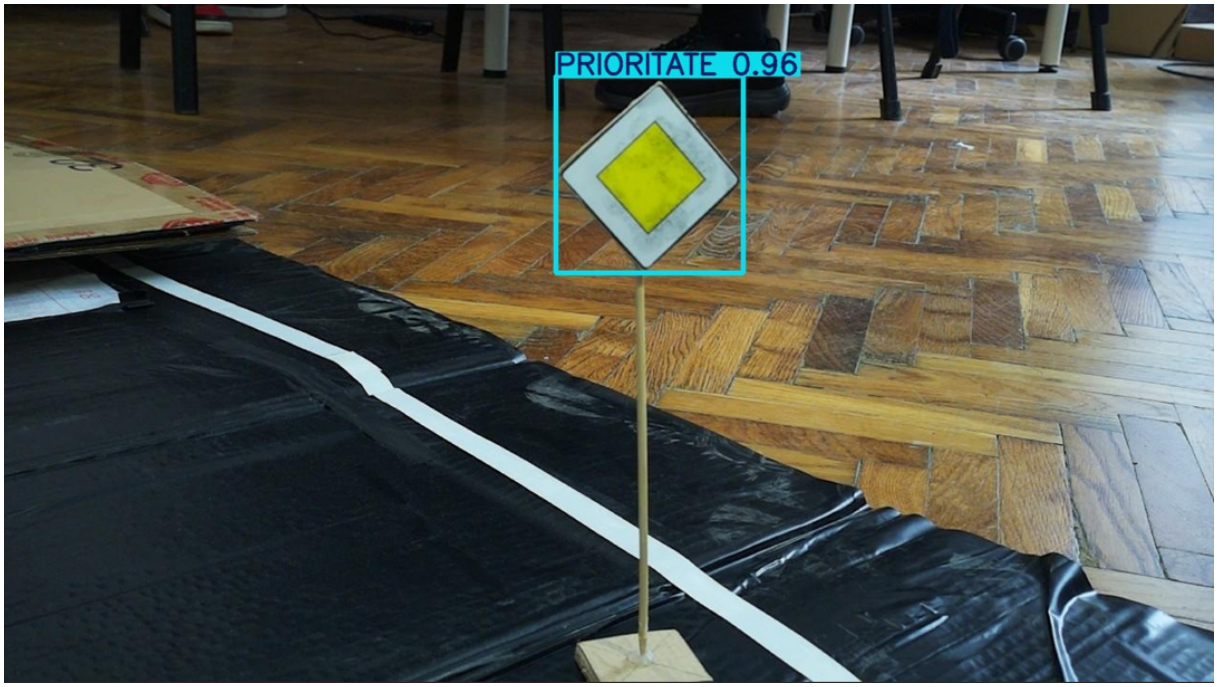


Figura 4.2: Exemplu de imagine prezisă pentru semnul PRIORITATE

Pentru a evalua performanța modelului, s-a folosit funcția **model.val**, aplicată asupra setului de testare definit în fișierul **data.yaml**. Rezultatele au fost structurate într-un tabel detaliat, care evidențiază performanțele obținute pentru fiecare clasă. Astfel, pentru cele patru clase definite s-au analizat mai mulți indicatori:

• **Instances** reprezintă numărul total de obiecte detectate din fiecare clasă în setul de test.

• **Precision (P)** arată proporția de obiecte detectate corect din tabelul obiectelor detectate. Valorile apropiate de 1 indică o precizie foarte bună.

• **Recall (R)** indică proporția obiectelor reale corect indentificate din totalul obiectelor reale prezente. Un recall ridicat indică o capacitate bună a modelului de a identifica toate obiectele relevante.

• **mAP50** este media preciziei calculate pentru pragul de intersecție peste uniune de 50% și reflectă performanța generală de detectare a modelului. Pragul de intersecție peste uniune (IoU) este o metrică esențială în evaluarea modelelor de detectare a obiectelor ce măsoară cât de bine se suprapune predicția modelului (dreptunghiul prezis) peste realitatea din imagine (dreptunghiul adnotat manual).

• **mAP50-95** este media preciziei calculate pentru mai multe praguri de IoU între 50% și 95%, oferind astfel o evaluare detaliată și exigentă.

Tabelul de mai jos oferă o privire în ansamblu asupra performanței algoritmului în recunoașterea semnelor de circulație. Se observă că toate cele patru clase definite au o precizie ridicată, valoarea fiind de peste 0.98 pentru fiecare clasă. De exemplu, STOP atinge o precizie perfectă de 1.000, ceea ce înseamnă că toate predicțiile pentru această clasă au fost corecte. De asemenea, clasa CEDEAZĂ TRECEREA are o precizie de 0.997 și un Recall de 1.000, ceea ce indică faptul că toate instanțele reale au fost identificate fără erori. Toate clasele au un scor mAP50 de 0.995, ceea ce indică o detecție precisă la un prag standard de 50% IoU, în schimb mAP50-95 variază ușor între 0.906 și 0.988, însă rămâne un scor ridicat, rezultatele sugerează că modelul reușește să identifice corect obiectele și să le localizeze cu mare acuratețe.

Clasă	**Imagini**	**Instanțe**	**Precizie (P)**	**Recall (R)**	**mAP50**	**mAP50-95**
CEDEAZĂ	12	12	0.997	1.000	0.995	0.95
PRIORITATE	12	12	0.984	1.000	0.995	0.957
STOP	33	33	1.000	0.978	0.995	0.988
TRECERE	16	16	0.992	1.000	0.995	0.948
Toate	70	73	0.993	0.994	0.995	0.906

Tabela 4.1: Performanța pe fiecare clasă detectată de YOLOv8

4.3 Organizarea modulară a sistemului

Din punct de vedere al arhitecturii software, întregul sistem este organizat modular, cu funcționalități distribuite între componentele hardware principale.

La pornirea telecomenzii, pe primul rând al ecranului LCD este afișat mesajul **Asistent ON**, însemnând că asistentul de conducere este conectat automat la pornirea sistemului. Acesta poate fi însă oprit, prin apăsarea butonului aferent joystickului din partea dreaptă a telecomenzii, iar mesajul afișat pe LCD se schimbă în **Asistent OFF**. Acesta poate fi comutat oricând, chiar și în timpul mersului, fără a fi necesară repornirea sistemului. Opțiunea selectată este transmisă wireless către vehicul prin intermediul modulelor NRF24L01, astfel încât asistentul va ști dacă e nevoie să intervină pe baza acestei valori.

Pentru a putea prelucra fluxul video provenit de la cameră, a fost necesară instalarea bibliotecii **libcamera**. Pe aceste imagini, modelul YOLOv8 antrenat anterior și încărcat din fișierul **best.pt** este rulat într-o buclă continuă. La detectarea unui semn de circulație, clasa corespunzătoare este transmisă imediat către placa Arduino Mega prin cablul USB A-B. În cazul în care se identifică mai multe semne simultan, este aplicată o regulă de prioritate, afișându-se doar semnul cel mai important, conform ordinii: STOP > TRECERE DE PIETONI > CEDEAZĂ TRECEREA > PRIORITATE. Arduino Mega primește clasa detectată și o transmite mai departe către telecomandă prin modulul său NRF24L01 configurat ca transmițător. Receptorul de pe telecomandă primește semnul și îl afișează pe al doilea rând al ecranului LCD, pentru a informa utilizatorul din timp despre semnul care urmează. Dacă timp de 2 secunde nu mai este detectat niciun semn, atunci rândul al doilea al ecranului este șters automat. Totuși, în cazul în care pe ecran apare un semn iar între timp este detectat alt semn cu o prioritate mai mare, semnul anterior este înlocuit instantaneu.

Mișcarea și direcția vehiculului sunt controlate de utilizator prin intermediul celor două joystickuri. Datele analogice citite de pe aceste joystickuri sunt mapate și trimise sub forma unui vector către mașină. Primul joystick controlează direcția de deplasare (înainte/înapoi), iar valorile pozitive sau negative determină sensul de rotație al motoarelor conectate la driverul L298N. Al doilea joystick reglează unghiul servomotorului responsabil pentru virare. Unghiul trimis este interpretat de biblioteca Servo.h, care acționează motorul între valorile de aproximativ 30 de grade (maxim stânga), 60 de grade (centru) și 90 de grade (maxim dreapta).

În cazul în care asistentul de conducere este activ, vehiculul se va opri imediat dacă unul dintre ce doi senzori cu infraroșu detectează banda de circulație sau în cazul în care utilizatorul nu oprește voluntar în apropierea semnului STOP sau la trecerea pentru pietoni. Pentru a nu confunda liniile albe ale trecerii de pietoni cu banda de circulație, valorile de la senzori vor fi ignorate dacă semnul trecerii de pietoni a fost detectat. În ambele cazuri de oprire forțată, un mesaj este trimis către telecomandă și afișat pe LCD.

Capitolul 5

Concluzii

Proiectul a reprezentat o provocare tehnologică complexă, atât din perspectiva construcției fizice a vehiculului, cât și din punct de vedere al dezvoltării software.

Una dintre cele mai mari dificultăți a fost proiectarea sistemului Ackermann în Fusion. Obținerea geometriei corecte a articulațiilor și a unghiurilor de virare necesare pentru ca roțile să se alinieze corespunzător a necesitat o perioadă semnificativă de testare și ajustare. Conectarea fizică a servomotorului la sistem a fost o altă etapă dificilă, întrucât a necesitat ajustarea tijei și modificarea poziției astfel încât unghiul de virare sa fie cât mai realist, fără blocaje sau limitări mecanice.

La nivelul comunicării între module, s-au întâmpinat obstacole importante. În fază inițială, fiecare componentă principală a sistemului (telecomandă și vehicul) dispunea de un singur modul NRF24L01, care era comutat software între modurile de recepție și transmisie. Totuși, această abordare nu a fost eficientă pentru că genera întârzieri semnificative ce afectau negativ timpul de reacție al vehiculului. Soluția identificată a constat în dublarea modulelor NRF, alocând câte un modul exclusiv pentru recepție și unul pentru transmisie atât pe mașină, cât și pe telecomandă. Această modificare a eliminat întârzierile și a dus la o comunicare bidirecțională eficientă și stabilă.

Obținerea unui model YOLOv8 performant a presupus, de asemenea, un efort susținut. Pentru a atinge valori ridicate de precizie, a fost necesară întocmirea unui set de date numeros și divers, iar în momentul rulării modelului, s-a observat o tendință a supraîncălzirii plăcii Raspberry Pi, ceea ce a impus integrarea unui ventilator suplimentar pentru a menține o temperatură stabilă.

Ca perspective de viitor, proiectul poate fi extins semnificativ. În primul rând, modelul de detecție poate fi antrenat cu un număr mult mai mare de clase pentru a recunoaște și alte semne de circulație. De asemenea, logica asistentului de conducere poate evolua pentru a lua decizii dinamice, nu doar oprirea vehiculului, ci adaptarea vitezei, schimbarea direcției sau evitarea obstacolelor în funcție de context.

Bibliografie

[1] Joe Walter, „The story of Ackermann steering”, în (2021), accesat în data de 30 mai 2025, URL: https://www.tiretechnologyinternational.com/opinion/the-story-of-ackermann-steering.html.

[2] „What is ADAS (Advanced Driver Assistance Systems)?”, în (2025), accesat în data de 30 mai 2025, URL: https://www.synopsys.com/glossary/what-is-adas.html.

[3] LawInfo Staff, „How Many Car Accidents Are Caused by Human Error?”, în (2024), revizuit de John Devendorf, Esq. Accesat în data de 30 mai 2025, URL: https://www.lawinfo.com/resources/car-accident/how-many-car-accidents-are-caused-by-human-er.html.

[4] „Average Car Dimensions: Length, Width & Height”, în (2025), accesat în data de 30 mai 2025, URL: https://www.icicilombard.com/blogs/car-insurance/mb/average-car-sizes-dimensions?.

[5] „Site-ul oficial Raspberry Pi”, în (2025), accesat în data de 4 iunie 2025, URL: https://www.raspberrypi.com/software/.

[6] „Site-ul oficial Visual Studio Code”, în (2025), accesat în data de 8 iunie 2025, URL: https://code.visualstudio.com/Download.

[7] „YOLOv8”, în (2025), accesat în data de 16 iunie 2025, URL: https://docs.ultralytics.com/models/yolov8/.

[8] „Ultralytics”, în (2025), accesat în data de 16 iunie 2025, URL: https://www.ultralytics.com.

[9] „Roboflow”, în (2025), accesat în data de 16 iunie 2025, URL: https://app.roboflow.com/horia-enescu-9echt.