Pregled upravljačkih sustava autonomnih vozila

Jurić, Kristijan

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI

Kristijan Jurić

PREGLED UPRAVLJAČKIH SUSTAVA AUTONOMNIH VOZILA

ZAVRŠNI RAD

Zagreb, 2019.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
ODBOR ZA ZAVRŠNI RAD

Zagreb, 29. ožujka 2019.

Zavod: **Zavod za inteligentne transportne sustave**
Predmet: **Automatsko upravljanje u prometu i transportu**

ZAVRŠNI ZADATAK br. 5098

Pristupnik: **Kristijan Jurić (0128056654)**
Studij: Inteligentni transportni sustavi i logistika
Smjer: Inteligentni transportni sustavi

Zadatak: **Pregled upravljačkih sustava autonomnih vozila**

Opis zadatka:

Autonomna vozila danas sve više ulaze u fazu testiranja, a neke razine autonomnosti su dostupne u komercijalnim vozilima. Pri tome su autonomnom vozilu potrebni različiti upravljački sustavi s pripadnim osjetilima. U ovom radu je potrebno opisati tehnologiju autonomnih vozila te napraviti pregled njihovih osnovnih upravljačkih sustava kao i objasniti glavne pristupe za izbjegavanje prepreka koji se koriste kod autonomnih vozila. Korištenjem Arduino razvojne platforme i postojeće makete malog cestovnog vozila izraditi simulator jednostavnog autonomnog vozila s mogućnošću izbjegavanja prepreka.

Mentor:

izv. prof. dr. sc. Edouard Ivanjko

Predsjednik povjerenstva za završni ispit:

Sveučilište u Zagrebu
Fakultet prometnih znanosti

ZAVRŠNI RAD

PREGLED UPRAVLJAČKIH SUSTAVA AUTONOMNIH VOZILA

AN OVERVIEW OF CONTROL SYSTEMS FOR AUTONOMOUS VEHICLE

Mentor: izv. prof. dr. sc. Edouard Ivanjko

Student: Kristijan Jurić
JMBAG: 0128056654

Zagreb, rujan 2019.

Sažetak

Naslov: Pregled upravljačkih sustava autonomnih vozila

Autonomna vozila, bila to cestovna, vodna ili zračna sve su više u primjeni, te se koriste u mnogobrojnim aplikacijama. Od autonomnih taxi službi do bespilotnih letjelica za nadgledanje određenih područja autonomna vozila sve više ulaze u područje svakodnevice. Kontinuiran tehnološki razvitak u posljednjih nekoliko desetljeća je omogućio autonomnim vozilima da postanu stvarnost, no problemi s dizajnom pouzdanih upravljačkih sustava koji bi omogućili učinkovito, korisno i najvažnije sigurno upravljanje autonomnim vozilom predstavlja prepreku.

U ovom završnom radu obrađuju se tehnologija i pristupi koji su se koristili u posljednjih 30 godina koji su i doveli do prvih autonomnih vozila, opisuju se standardni upravljački sustavi autonomnih vozila današnjice, osnovni pristupi za izbjegavanje prepreka u autonomnoj vožnji, te se dokazuje funkcionalnost samog rada takvog upravljačkog sustava uporabom Arduino razvojne platforme u sukladnosti sa različitim senzorima implementiranih na postojećoj maketi malog cestovnog vozila, pripadajućim sklopovljem, napisanim programskim kodom i dijagramom toka koji i upravljaju radom simulatora autonomnog vozila. Simulacija rada autonomnog vozila se vrši na konstruiranom poligonu, gdje se u stvarnom vremenu očitavaju podaci sa senzora i prosljeđuju se dalje procesnoj jedinici na obradu koja prema definiranoj logici donosi odluke u vožnji.

Ključne riječi: Arduino; simulator autonomnog vozila; upravljački sustavi; pouzdanih

Abstract

Title: An overview of control systems for autonomous vehicle

Autonomous vehicles, whether be road, water or air, are increasingly in use in many applications. From autonomous taxi services to unmanned aerial vehicles for surveillance of certain areas autonomous vehicles are increasingly entering the realm of everyday life.The continuous technological development in the last few decades have made autonomous vehicles to become a reality, but problems with the design of reliable control systems which would allow for an efficient, useful and most importantly safe operation of an autonomous vehicle is an obstacle.

This final paper addresses the technology and approaches used in the the last 30 years that led to the first autonomous vehicles, standard control systems of an autonomous vehicle is described, main approaches for obstacle avoidance in autonomous driving and also the functionality of workable control system using an Arduino development platform is proven in accordance with various sensors implemented on an existing model of a small road vehicle, with associated hardware, written code and flowchart that control the operation of the autonomous vehicle simulator. The operation of an autonomous vehicle is simulated on a constructed polygon course, where real-time sensor data is read and forwarded to a processing unit that makes driving decisions by defined logic.

Keywords: Arduino; autonomous vehicle simulator; control systems; reliable

Sadržaj

1 Uvod

Vozila su jednom bila smatrana područjem koje je bilo rezervirano samo za mehaničke inženjere, no stalnim proučavanjem polja umjetne inteligencije te sve većim napretkom po pitanju istog je pretvorilo standardno vozilo u inteligentno, informacijama bogato prijevozno sredstvo kojem nije potrebno ljudsko upravljanje za prometovanje. U svrhu smanjenja prometnih gužvi, emisije stakleničkih plinova i drugih zagađivača zraka, te povećanja mobilnosti starijima osobama, osobama sa invaliditetom i smanjenom pokretljivošću potrebno je razmotriti nove tehnologije, odnosno Inteligentne Transportne Sustave (ITS). ITS je grana koja se bavi nadogradnjom klasičnog prometnog i transportno-logističkog sustava informacijsko-komunikacijskim elementima sa svrhom povećanja sigurnosti, protočnosti, boljem informiranju putnika te reduciranju vremena putovanja.

Pod dio ITS-a se podrazumijeva i autonomno-inteligentno vozilo čija se funkcionalnost ostvaruje putem telematičke opreme koja se nadograđuje na standardnu opremu vozila, te isti moraju biti precizni, pouzdani te izvedeni na način da vozilo može točno,lako, brzo i na siguran način mijenjati smjer.

U ovom završnom radu bit će opisana tehnologija autonomnih vozila, njihovi upravljački sustavi, tehnike i pristupi kod izbjegavanja prepreka, te će se pomoću Arduino razvojne platforme i makete malog cestovnog vozila izraditi simulator jednostavnog autonomnog vozila s mogućnošću izbjegavanja prepreka. Opći cilj rada je opisati razvoj i tehnologiju autonomnog vozila kroz povijest, upravljačke sustave bez kojih vozilo nebi moglo samo razmišljati niti primati parametre u realnom vremenu od senzora sto omogućava autonomnost. Specifični cilj je prikazati njihov rad na primjeru makete jednostavnog sustava autonomnog vozila izrađene na bazi mikrokontrolera i mogućnosti primjene istog. Naslov završnog rada je: Pregled upravljačkih sustava autonomnih vozila. Rad je podijeljen u sedam cjelina:

1. Uvod;

2. Povijest razvoja i tehnologija autonomnih vozila;

3. Glavni sustavi upravljana autonomnih vozila;

4. Pristupi za izbjegavanje prepreka;

5. Sklopovska i programska podrška simulatora autonomnog vozila;

6. Testiranje simulatora autonomnog vozila;

7. Zaključak.

U drugom poglavlju, opisan je tijek razvoja i tehnologije autonomnih vozila gdje se najviše spominju ključni trenutci u povijesti koji su imali veliki značaj za razvoj autonomnosti. Tijek razvoja se podijelio u 3 ključna razdoblja: fazu temeljnog istraživanja gdje su se testovi izvodili najviše na autocestama zbog smanjene kompleksnosti kod prepoznavanja krivine, fazu velikih izazova obilježenu natjecanjem organiziranim od strane Agencije za obrambene napredne istraživačke projekte američkog Ministarstva obrane (DARPA) što je ubrzalo razvoj tehnologije autonomnih vozila, te fazu komercijalnog razvoja obilježenu stvaranjem različitih razvojnih grupa, suradnjom već postojećih, te ulaskom autonomnih vozila u komercijalne vode. Kako bi autonomno vozilo moglo bez problema izbjegavati prepreke ili raspoznavati znakove potrebno su upravljački sustavi koji su opisani u trećem poglavlju. Razmatrat će se o njihovim tehničkim karakteristikama, načinu upotrebe, te primjeni u autonomnoj vožnji.

Kod izbjegavanja vozila ili objekata potrebno je uključiti odgovarajuću softversku podršku koja će u realnom vremenu koristeći različite algoritme i funkcije omogućiti autonomnom vozilu da ih pravovremeno izbjegne, te najosnovniji pristupi koje autonomna vozila koriste za izbjegavanje su opisani u četvrtom poglavlju. Za realizaciju završnog rada u potpunosti iskorištena je razvojna platforma Arduino koja sa ulazno/izlaznim pinovima uspostavlja komunikaciju sa različitim osjetilima kao što su ultrazvučni senzori, infracrveni senzori ili drugim mikrokontrolerima kao žiroskopom, te sa odgovarajućom programskom

podrškom primljene podatke iz osjetila obrađuje i djeluje na temelju rezultata same obrade. Kako bi kontrolni sustav simulatora autonomnog vozila funkcionirao potrebno je napraviti sklop čija je pojednostavljena električna shema prikazana i objašnjena u petom poglavlju. Isto tako će se proći kroz programski kod i dijagram toka, te će time biti jasno na koji način simulator jednostavnog sustava autonomnog vozila funkcionira. U šestom poglavlju je opisano testiranje simulatora na malom poligonu i dobijeni podaci prema kojima model autonomnog vozila određuje daljni smjer kretanja.

Rad završava zaključkom te prijedlozima za nastavak rada na ovoj temi.

2 Povijest razvoja i tehnologija autonomnih vozila

Evolucija i samo pojavljivanje autonomnih vozila je nastalo kao rezultat neprestanih istraživanja iz područja bežičnih mreža, integriranih sustava, navigacije, testiranja senzora, mrežnih tehnologija, te same tehnike analize i obrade prikupljenih podataka.

Sam pojam autonomnih vozila prvi put se pojavljuje u testu kojeg su izveli inženjeri Radio Test Service-a 1921. godine gdje se vozilo kontroliralo radio signalima iz vojnog kamiona koji je tijekom samog testa vozio 30 m iza. Jedan od drugih primjera je auto "American Wonder" iz 1925.godine nastao u Sjedinjenim Američkim Državama gdje se vozilo opremljeno antenama koje su primale radio signale odašiljane sa drugog vozila kontroliralo također pomoću daljinskog upravljača [1]. Iako sva ova vozila nisu bila zapravo autonomna nego daljinski upravljana važno ih je spomenuti jer su samo jedan od primjera pionirskih projekata koji su doveli do prvog autonomnog vozila.

U slijedećim potpoglavljima opisana su tri glavna perioda tehnoloških dostignuća postignutih u posljednjih 30 godina u području autonomnih vozila.

2.1 Faza temeljnog istraživanja

Sveučilišna istraživačka odjeljenja su u periodu od 1980. do 2003. najčešće u suradnji sa agencijama za transportne usluge, te auto kompanijama imala osnovne studije iz autonomnog prijevoza. Iz tih perioda istraživanja su proizašla dva glavna tehnološka koncepta. Kao poticaj sa jedne strane, istraživači su krenuli prema izradi i proučavanju autonomnih sustava prilagođenih uvjetima na autocestama u kojima vozila najviše ovise o samoj infrastrukturi autoceste koja im služi za usmjeravanje. Jedan od takvih sustava prvi put je demonstriran 1997. godine na 12 kilometara dugoj Kalifornijskoj autocesti pored San Diego-a. Projekt zvan "DEMO97" demonstrirao je autonomnu vožnju osam vozila

usmjeravanih magnetima koji su bili ugrađeni u samu autocestu, te koordinirani sustavom komunikacije vozila sa drugim vozilom (V2V) [1]. V2V je automobilska tehnologija koja omogućuje automobilima da "razgovaraju" jedni sa drugima primjenom računalnih mreža u kojima vozila povezana sa komunikacijskim jedinicama postavljenim pored ceste informiraju jedno drugo o opasnostima ili stanju u prometu.

Slika 1: Kombi marke Mercedes-Benz redizajniran kao autonomno vozilo od strane tima Ernst Dickamnn-a

Sa druge strane istraživačke grupe su bile potaknute u proizvodnji polu-autonomnih i autonomnih vozila koja su malo ovisila o samoj infrastrukturi autoceste. Početkom 1980-te Ernst Dickmann koji se smatra pionirom automobilske robotike je sa svojim timom sa Sveučilišta Bundeswehr u Munich-u razvio autonomno vozilo vođeno računalnim vidom koje se kretalo brzinom od 100 kilometara na sat bez prometa. Kombi marke Marcedes-Benz prikazan na slici 1 opremljen kamerama i drugim senzorima bio je redizajniran kako bi bilo moguće kontrolirati upravljač, gas i kočnice uz pomoć računalnih naredbi koje su djelovale na temelju obrade slike u realnom vremenu [1]. Istovremeno od sredine 1980-ih do početka 2000.godine razvojna grupa NavLab sa Sveučilišta Cornegie Mellon je razvila seriju od 11 vozila nazvanih Navlab sa na kraju imena dodijeljenim rednim brojem. Godine 1995. NavLab 5 vozilo je odvozilo rutu od 4586 kilometara koja se prostirala od Pittsburgh-a do San Diego-a u kojem je vozilo samostalno upravljalo 98% vremena cijelog putovanja dok su dodavanjem gasa i kontrolom kočnica upravljali ljudski operatori [2].

2.2 Faza velikih izazova

U periodu od 2003. godine do 2007. godine Agencija za obrambene napredne istraživačke projekte američkog Ministarstva obrane (DARPA) je održala tri velika natjecanja koja su iznimno ubrzala razvoj tehnologije autonomnih vozila, te isto tako su usmjerila oči javnosti na važnost i korisnost istog [1]. Glavni cilj je bio izdvojiti najbolje projekte te vidjeti dali postoji mogućnost primjene autonomnih vozila u vojne svrhe.

Na prvom natjecanju održanom u ožujku. 2004. godine nitko nije mogao završiti dodijeljenu rutu koja se prostirala od grada Bastow-a u Kaliforniji do grada Primm-a u Nevadi. Organizacija je nudila nagradu u iznosu od milijun dolara skupini od petnaest finalista koji su uspjeli proći kvalifikacije na "California Speedway" traci, no nagradu na kraju nije osvojio nitko zbog toga što je dodijeljena ruta bila pre-kompliciranа za savladat. Nakon 18 mjeseci DARPA je organizirala drugo natjecanje kojeg je pet grupa natjecatelja uspješno savladalo s obzirom na dodijeljenu rutu. Najbrža grupa je savladala rutu u nešto manje od sedam sati, dok sva sljedeća vozila od pet uspješnih u narednih 35 minuta nakon najbržeg [2].

DARPA je održala svoje treće i zadnje natjecanje 2007.godine nazvano "Urban Challenge" gdje su vozila morala savladati rutu u urbanom okruženju duljine od 96 kilometara pridržavajući se prometnih pravila, te vozeći sukladno sa drugim autonomnim i ne-autonomnim vozilima [1]. Rutu je završilo šest grupa, a tri su završile utrku u vremenu od četiri i pol sata gdje su se uključivale i kazne zbog kršenja prometnih i sigurnosnih pravila.

Veliko natjecanje "Grand Challenge" pokrenuto od strane DARPA-e je ubrzalo razvoj senzorskih sustava i računskih algoritama za detekciju i reakciju na ponašanje drugih vozila, te njihove sustave praćenja obilježenih cesta i mogućnosti da prate prometnu signalizaciju i pravila.

2.3 Faza komercijalnog razvoja

Natjecanja koju su bila održana su uspostavila suradnju između auto-industrije i edukacijskog sektora što je donijelo i niz resursa koja su omogućila autonomnom sektoru brži napredak u razvoju autonomnih vozila. To uključuje i stvaranje Autonomous Driving

Collaborative Research Lab razvojne grupe, suradnje između General Motors-a i Carnegie Mellon Sveučilišta i isto tako suradnjom između auto kompanije Wolkswagen i Sveučilišta Stanford.

Slika 2: Lexus RX450h redizajniran za "Google Driverless Car" program

Razvojni program započet nedugo nakon zadnjeg "Grand Challenge" natjecanja koji je okupio najbolje inženjere i istraživače iz različitih razvojnih grupa koji su sudjelovali u istom natjecanju je bio "Google's Driverless Car" program. Razvoj se odvijao u tajnim X laboratorijima kojeg je vodio Sebastian Thrun, bivši direktor "Stanford Artificial Intelligence" laboratorija, te ko-izumitelj Google Street View-a [3].

Grupa Sebastiana Thruna je isto tako i osvojila "2005 DARPA Grand Challenge" nagradu za svoje autonomno vozilo "Stanley" te su s time mogli lagano preći na zahtjevnije projekte koji su doveli do današnjih modernih "Waymo1" autonomnih vozila [3]. Na slici 2 je prikazan Lexus RX450H automobil koji je bio opremljen opremom za autonomno upravljanje, te je jedan mnogi projekata koji su proizašli iz programa "Google Driverless Car". Google-ov interes za razvijanjem u tom području je omogućio lagani prijelaz programa autonomnih vozila iz istraživačkog sektora u komercijalni sektor. Kako Google nije djelovao sam u tom istraživačkom području, 2013. godine Audi i Toyota su predstavili svoje programe računalnog vida autonomnih vozila, te druge istraživačke programe na "International Consumer Electronics Show" konferenciji [2].

3 Glavni sustavi upravljanja autonomnih vozila

Ideja izrade u potpunosti pouzdanih autonomnih vozila nije nova, te njezinu materijalizaciju je onemogućilo nedovoljno brzi razvitak tehnologije iz tog područja. Iako se u posljednjem desetljeću mnogo napravilo u vezi razvoja algoritama, software-a i hardware-a i dalje postoje izazovi koje se treba savladati kako bi upravljački sustavi autonomnih vozila funkcionirali savršeno. Prije samog razmatranja kontrolnih sustava upravljanja potrebno je navesti klasifikaciju razina autonomnih cestovnih vozila jer o njima zapravo ovisi kako će upravljački sustav biti izveden. Različiti stupnjevi autonomnosti su prvo bili izdani od strane National Highway Safety Administration-a (NHTSA) 2013. godine, te godinu dana kasnije je Society of Automotive Engineers izdao klasifikaciju SAE. Society of Automotive Engineers je postavio klasifikaciju u 6 razina [4]:

1. RAZINA 0 - to je tzv. ne-autonomna razina, u ovoj razini vozač potpuno samostalno upravlja vozilom;

2. RAZINA 1 - autonomija na ovoj razini uključuje automatizaciju ponekih parametara vožnje; primjer su slučajevi u kojem sustav ima kontrolu kočenja ili ubrzavanja;

3. RAZINA 2 - na ovoj razini barem dva parametara moraju biti u potpunosti automatizirana i rade kooperativno kako bi vozača oslobodili od upravljanja tim funkcijama;

4. RAZINA 3 - vozila ove razine prikupljaju podatke iz okoline, stvaraju putanju za praćenje i indentificiraju prepreke. Vozač ima mogućnost obavljati druge funkcije no i dalje mora biti oprezan u slučaju ako je njegova intervencija potrebna;

5. RAZINA 4 - vozilo visoke razine automatizacije; vozilo je dizajnirano na način da samostalno izvršava zadatke gledajući sa stajališta operacijskog aspekta (upravljanje, kočenje, ubrzavanje itd.), taktičkog aspekta (promjena vozne trake, davanje signala itd.), te reakcija na opasnost; Vozilo može zatražiti interakciju vozača u slučaju iznimno kompleksnih situacija;

6. RAZINA 5 - potpuno autonomno vozilo; vozilo funkcionira na način da vozač mora samo unijeti željenu rutu, te ostale funkcije sa maksimalnom razinom pouzdanosti vozilo samo obavlja. Ovakva vozila mogu prometovati i bez prisustva vozača.

Neovisno o marki i modelu arhitektura upravljačkog sustava autonomnog vozila petog stupnja autonomnosti prema SAE klasifikaciji je već definirana, te je podijeljena u četiri dijela koja se sastoji od senzorskih sustava, klijentskih sustava, akcijskih sustava i korisničkih sustava koji će biti opisani u sljedećim poglavljima.

3.1 Senzorski sustavi

Senzorski sustavi igraju ključnu ulogu u autonomnoj vožnji, te su sastavljeni od različitih senzora koji su odgovorni za prikupljanje podataka iz okoline u realnom vremenu. Senzori kod upravljačkih sustava autonomnih vozila se mogu podijeliti u dvije grupe i to na senzore kratkog dometa i srednjeg/dugog dometa. Senzori kratkog dometa su ultrazvučni senzori, kapacitativni senzori ili infracrveni senzori dok senzori dugog dometa su radar, LIDAR, računalni vid i GPS [5]. Jedna od veliki mana senzora dugog dometa je što se često njihova izvedba nepredvidljiva u nekim situacijama zbog dinamičnosti okoline, te je zato iznimno važno autonomno vozilo opremiti sa više tipova senzora koji će sa "fuzijom" senzorskih podataka komplementirat jedno drugome. U sljedećim pod-poglavljima se opisuju jedni od važnih senzora srednjeg/dugog dometa kojima moraju biti opremljena sva autonomna vozila petog stupnja prema SAE kvalifikaciji:

3.1.1 Radar

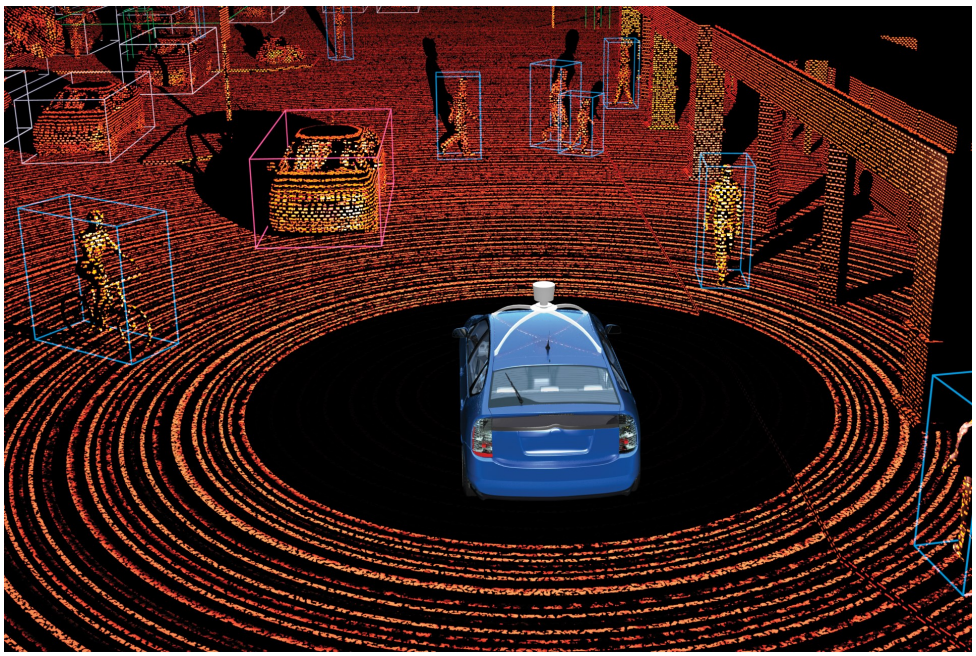
Radar dolazi od engleske riječi Radio Detection and Ranging, te predstavlja elektronički uređaj koji služi za određivanje udaljenosti teško uočljivih, nevidljivih predmeta ili objekata [6]. Na radar ne utječu vremenske nepogode i ima mogućnost otkrivanja objekata po mraku, magli ili drugim uvjetima smanjene vidljivosti. Jedno od najvažnijih područja u kojem se koristi radar je kontrola zračnog prometa, te se tamo koriste promatrački radari velikog dometa i radari koji nadziru sam prilaz zračnoj luci.

U autonomnom vozilu radari su postavljeni po cijelom vozilu tako da pokrivaju cijelo prednje područje ispred vozila, te funkcioniraju na način da šalju ultrakratke

elektromagnetske valove koji se mogu reflektirati od krute površine npr. stražnjeg dijela drugog vozila i šalju se natrag [5]. Nakon što radar zaprimi reflektirani signal on dobija informaciju koliko je zapravo objekt ili prepreka udaljena od radara i koliko se brzo kreće, te time prati brzine drugih vozila u stvarnom vremenu koji ga okružuju.
Radar koji se koristi kao vizualni senzor u autonomnim vozilima obuhvaća frekvencijsko područje od 75 do 110 GHz, te valnu duljinu od 2,7 do 4 mm [6].

3.1.2 LIDAR

LIDAR je skraćenica od engleske riječi Light Detection and Ranging, te predstavlja optički mjerni instrument koji odašilje laserske zrake koje se odbijaju od vrlo sitnih objekata ili čestica. Sastoji se od laserskog uređaja velike preciznosti koji može precizno očitavati čak i u jako nepogodnim vremenskim uvjetima, te on osvjetljava česticu nekog objekta sa laserskim svjetlom, mjeri vrijeme koliko je potrebno da se signal reflektira i na taj način izračunava udaljenost od tog objekta ili čestice.



Slika 3: Primjer snimanja okoline LIDAR senzorom

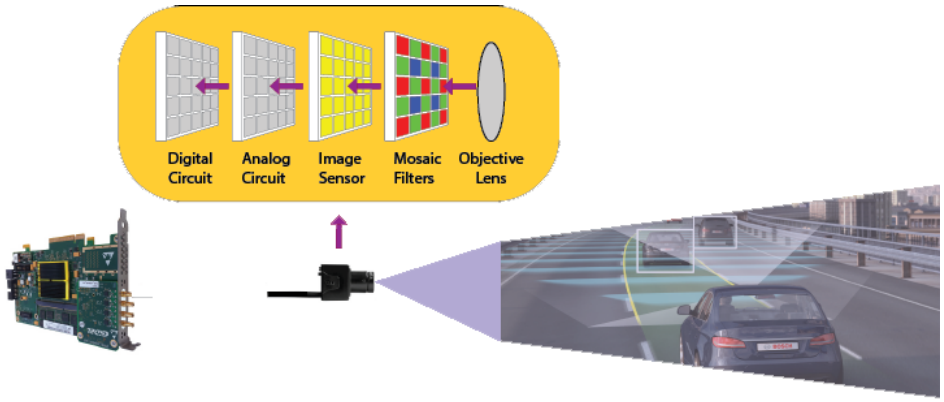
Prilikom prepoznavanja objekta LIDAR stvara detaljnu 3D mapu terena prikazanog na slici 3 sa kojom vozilo može razlikovati automobile, kamione, bicikle itd. i ovisno o tome reagirati na odgovarajući način. Jedan od glavnih problema LIDAR-a je to što on

sa skeniranjem dobija reljefnu sliku okoline u jednoj cjelini što i dalje nije dovoljno da sa velikom pouzdanosti diferencira te iste različite objekte. Standardni LIDAR rotira na frekvenciji od 10 Hz i prima 1.3 milijuna očitanja u sekundi [6].

3.1.3 Računalni vid

Računalni vid znanstvena je i tehnološka disciplina koja se bavi teorijom i izradom samog sustava koji služe dobivanju informacija iz slika, bilo to iz jedne ili više fotografija, video uradaka ili određenih medicinskih uređaja. Uz to cilj računalnog vida je prepoznavanje objekata, praćenje objekata, detekcija unaprijed zadanih događaja, rekonstrukcija slike i sl. [7]

Za realizaciju računalnog vida kod autonomnih vozila koristi se osam ili više kamera koje služe kako bi prepoznavale objekte, pratile putanju vozne trake, svjetlosne signale itd. Kamere kao senzori prikupljaju mnogo više vizualnih informacija i time mnogo bolje prate okolinu od drugih senzora. Jedna od glavnih nedostataka kamera je to što su osjetljive na smetnje kao bljesak svjetlosti pri izlasku iz tunela na sunčan dan i općenito na nepogodne vremenske uvjete dok obrada podataka zahtjeva puno računske snage [8].



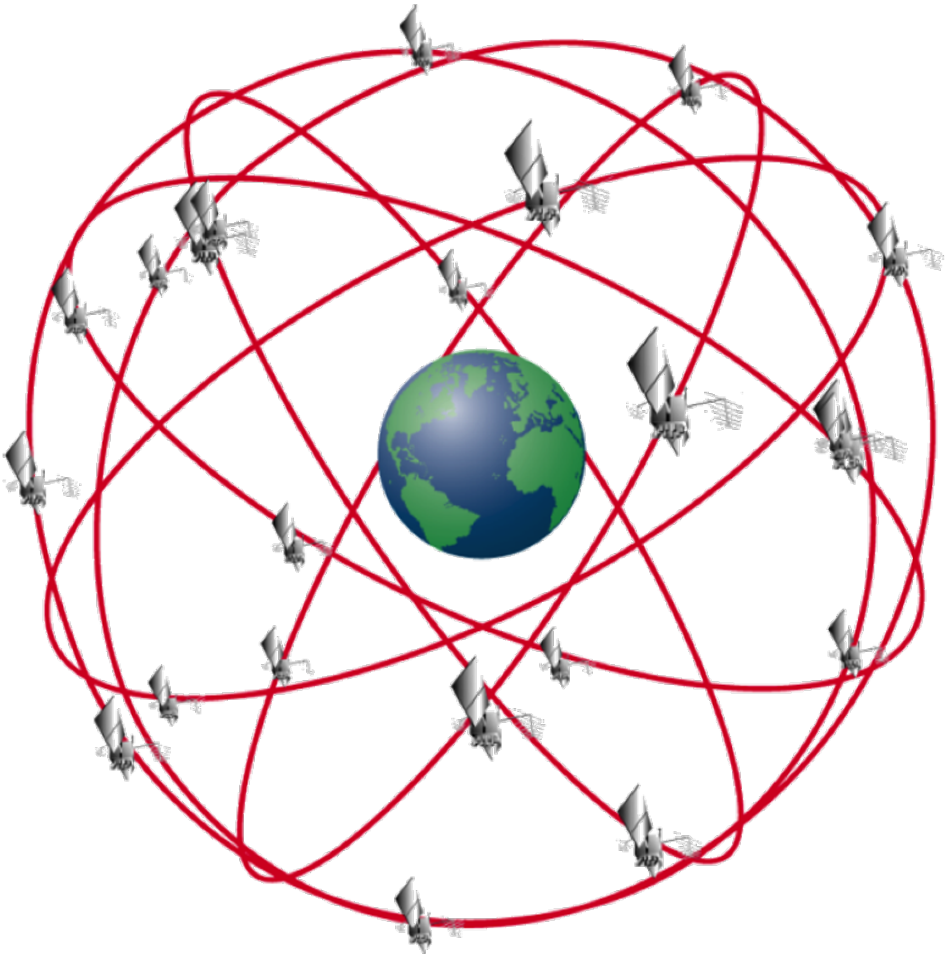
Slika 4: Princip rada računalnog vida

Na slici 4 prikazan je princip funkcioniranja sustava računalnog vida. Kamera preko svoje fokusne leće i matrice foto-detektora za svaki piksel koji se nalazi u vidnom polju šalje informaciju o stanjima piksela (0-255) jedinici za procesiranje vizualnih informacija. Procesna jedinica vrši obradu zaprimljenih informacija i detektira objekte koristeći određene algoritme kao SIFT (Scale-Invariant Feaeture Transform) ili Viola-Jones koji se često

koristi za uočavanje pješaka, automobila, prometnih znakova, pješačkih prijelaza i sl [7].

3.1.4 GPS

GNSS je internacionalni naziv za mrežu satelita namijenjenih navigaciji i pozicioniranju u koju spadaju GPS, GLONASS, Galileo i drugi. GPS ili Globalni pozicijski sustav (engl. Global Positioning System) temelji se na satelitima koji neprekidno kruže oko Zemlje te tako osiguravaju točnu lokaciju i poziciju u bilo koje doba dana. Potreban je u autonomnim vozilima zato što određuje rutu kojom će vozilo putovati od trenutka kada korisnik unese željenu lokaciju sve do dolaska na samo odredište. Sastoji se od svemirskog, kontrolnog i korisničkog segmenta. [9]



Slika 5: Konstelacija GPS satelita

Svemirski segment GPS-a prikazanom na slici 5 čine 32 satelita koji se gibaju oko zemlje u kružnim orbitama na visini od oko 20000 km. Kako bi GPS sateliti pokrivali što

veće područje potrebno je da se uvijek iznad horizonta nalazi barem pet satelita. Globalna pokrivenost GPS navigacijskog sustava se postiže sa samom strukturom njihovog rasporeda u 6 orbitalnih ravnina koje imaju inklinaciju (nagib prema Zemljinom ekvatoru) od 55° te su razmaknute za 60°. Ophodno vrijeme satelita iznosi 12 sati, te se kreću brzinom od 11 000 km/h. [9]

Kontrolni segment se sastoji od pet kontrolnih stanica diljem svijeta koje obavljaju nadzor, prate i upravljaju satelitima. Glavna kontrola stanica se nalazi u Colorado Springs-u, te ona izračunava odstupanje svakog satelita pa korekcije šalje natrag satelitima nekoliko puta dnevno dok ostale četiri kontrolne stanice nemaju ljude u sebi nego one informacije dobivene od satelita prosljeđuju dalje glavnoj kontrolnoj stanici.

Korisnički segment sačinjavaju dvije kategorije korisnika: autorizirani i neautorizirani korisnici. Autorizirane korisnike čini vojska i državne službe dok neautorizirane korisnike čine svi drugi korisnici diljem svijeta. [6]

Princip rada sustava temelji se na visokom stupnju stabilnosti atomskih „satova“ ugrađenih na satelite i na iznimno visokoj točnosti mjerenja vremena. Sustav radi u vlastitom vremenu pa nije potrebna stalna sinkronizacija. Kako bi GPS prijemnik mogao odrediti svoju poziciju, treba izmjeriti udaljenost do satelita, znati poziciju satelita i znati podatke za uračunavanje korekcije.

Korisnost samog GPS sustava kod autonomnih vozila je velika bez obzira što često signal zna biti prekinut zbog opstrukcije objekata kao što su zgrade, tuneli i guste šume. Mana GPS-a je što mu treba dugo vremena kako bi se ponovo ažurirao što nepoželjna stavka kada se on koristi u stvarnom vremenu.

3.2 Klijentski sustavi

Klijentski sustav općenito predstavlja mozak autonomnog vozila koji ima zadaću da obrađuje sve prikupljene podatke i izdvaja najvažnije informacije kako bi se interpretirala okolina, odredio položaj samog vozila, te isto tako odlučilo o sljedećim koracima u samoj vožnji (odlukama).

Percepcija ili interpretacija okoline se sastoji od tri dijela: pozicioniranja, detekcije i praćenja. Kako bi sve sve to realiziralo mora se napraviti spoj ili "fuzija" podataka primljenih od strane različitih senzora. Spoj prikupljenih podataka se izvodi preko

algoritama koji rade na tri različite razine: niskoj, srednjoj i visokoj. Tek na srednjoj razini dolazi do glavne fuzije senzorskih podataka, te na visokoj razini se uzimaju različite donesene odluke bazirane na podacima pojedinog senzora i na temelju njih se stvara konačna odluka koja se šalje akcijskom sustavu na realizaciju.

3.3 Akcijski sustavi

Akcijski sustavi predstavljaju mehaničke dijelove autonomnog vozila kao upravljač, kočne i pogonske sustave itd. koji zapravo i izvode naredbe koje su zaprimljene od klijentskog sustava, te usmjeravaju autonomno vozilo.

3.4 Korisnički sustavi

Korisnički sustavi su kombinacija hardware-a i software-a koji omogućuju korisniku autonomnog vozila da komunicira sa njime u stvarnom vremenu. Interakcijom s korisničkim sustavom putnik može dobijat informacije o kvaliteti, performansama vožnje, te također može zatražiti od autonomnog vozila da obavlja određene zadatke unoseći zahtjeve od strane vozila [5].

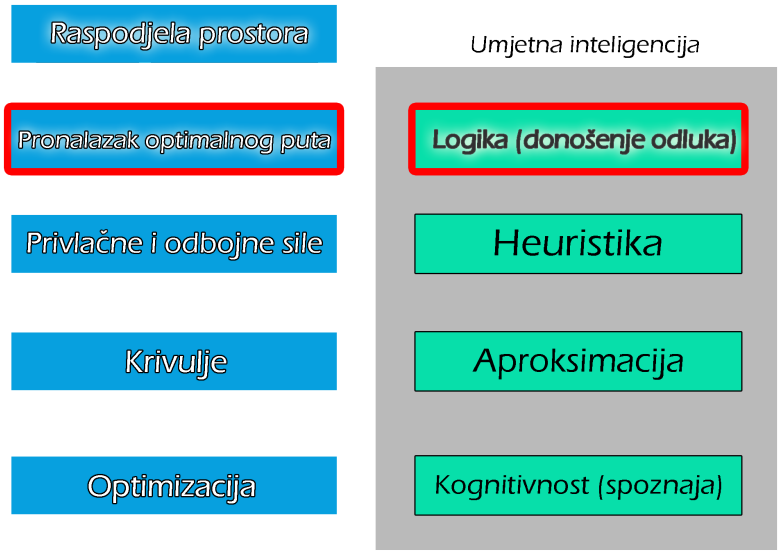
Elementi korisničkog sustava se sastoje od pokazivača koji daju informaciju korisniku o ruti vozila ili sljedećim radnjama koje namjerava izvršiti. Putnici dobijaju informacije o okolini oko autonomnog vozila, te druge oblike informacija vezane za uvjete unutar vozila kako bi se korisnik osjećao ugodnije i samopouzdanije prilikom njegovog korištenja.

Moderni koncepti korisničkog sustava autonomnog vozila namjeravaju maknuti standardne elemente kao što su volan, pedale i mjenjač, te ih zamijeniti jednim velikim ekranom koji će pokazivati informacije o okolini, te će samo dopuštati korisniku da ga pokrene, zaustavi i unese željenu rutu. Sve druge interakcije sa sustavima zabave ili informacijskim sustavima bi bile rađene preko mobilnog uređaja.

4 Pristupi za izbjegavanje prepreka

Kod modernih autonomnih vozila koriste se različiti pristupi kod izbjegavanja prepreka koji ovise o ulaznim podacima zaprimljenih od strane različitih senzora koji omogućuju stvaranje percepcijske mape sa objektima, voznim trakama, prometom , zakrivljenošću ceste i slično. ovisno o kojima vozilo planira svoj trenutni i sljedeći korak u vožnji.

Kako bi se bolje razumio sam princip odabira kretanja potrebno je spomenuti tri glavne grupe vještina i kontrola koje autonomno vozilo posjeduje: strateške kontrole (planiranje rute, predviđanje, odluke itd.), taktičke kontrole (promjena vozne trake, davanje signala itd.), te operacijske kontrole (upravljanje, kočenje i ubrzavanje) [4]. Operacijske kontrole su one koje zapravo i izvode samo izbjegavanje, no potrebno je prvo sa strateškog aspekta definirati rutu, ovisno o položaju prepreka isplanirati primjenom različitih algoritama rutu izbjegavanja, te onda informacije poslati taktičkim kontrolama koje planiraju niže razine funkcija vozila.



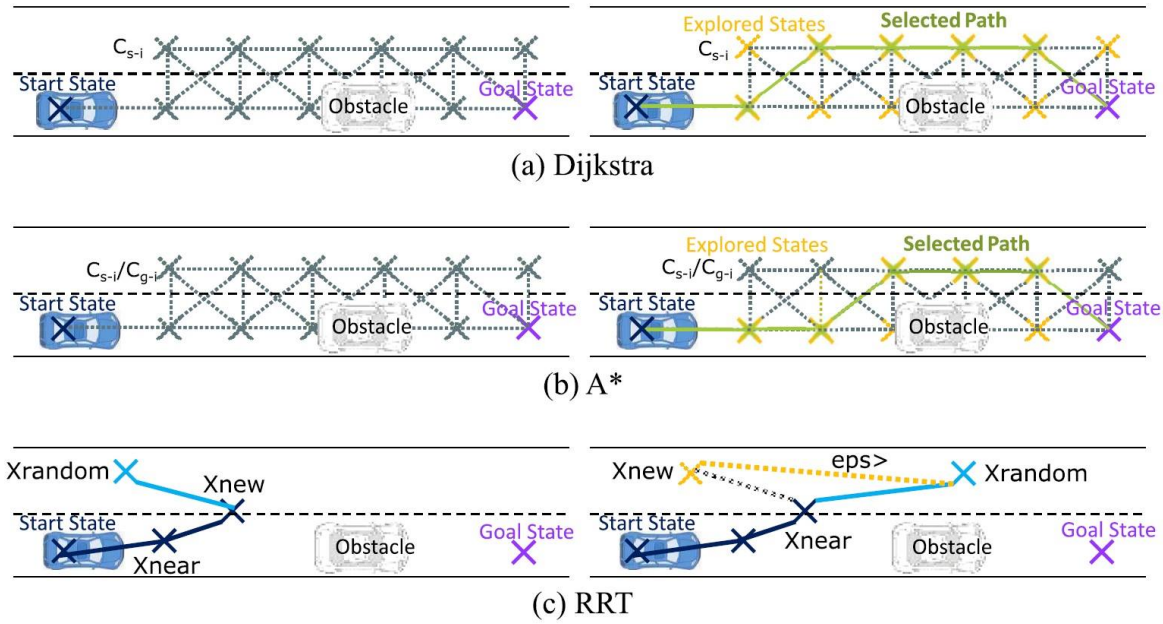
Slika 6: Podjela pristupa kod izbjegavanja prepreka

Pristupi kod izbjegavanja objekata i općenitog planiranja kretanja vozila su podijeljeni u različite grupe prikazane na slici 6, te su jedno od popularnih pristupa koji se koriste jos od natjecanja organiziranih od strane DAPRA-e 2007.godine. Obitelj pristupa koji su zapravo algoritmi koji rade na istom principu vraćaju isti tip rezultata na izlazu kao i drugi algoritmi te obitelji, kvalificirani istim značajkama i matematičkim poljima.

U sljedećim pod-poglavljima se opisuju najjednostavniji pristupi iz te obitelji koji se primjenjuju kod izbjegavanja objekata i planiranja kretanja mobilnih robota i autonomnih vozila izdvojeni iz razloga velike širine polja proučavanja.

4.1 Pronalazak optimalnog puta

Pristup pronalaska optimalnog puta je pod-dio teorije grafova koji se primjenjuju u operacijskim istraživanjima kako bi se rješavali kombinatorni problemi u sukladnosti sa grafičkim prikazom. Graf može biti težinski opterećen i orijentiran sa uzorcima točaka, ćelija, ili čvorovima. Glavni princip je da se nađe put u grafu kako bi se optimizirala funkcija troška. Prijeđena udaljenost, potrošnja goriva i udobnost su glavne varijable funkcije troška. [4]



Slika 7: Algoritmi korišteni u pristupu pronalaska optimalnog puta

Dikstrin algoritam predstavlja metodu kojom se pronalazi najkraća cjelokupna uda-

ljenost između dvije točke, ćelije ili čvora, te se često koristi za planiranje pokreta kod autonomnih vozila. Algoritam traži i provjerava sve smjerove mogućeg kretanja prikazano na slici 7(a), te pronalazi optimalan put u odnosu sa iznosima varijabla funkcija troška.

Nedostaci Dikstrinog algoritma su duga vremena procesiranja, te su djelomično uklonjeni primjenom A* algoritma. Primjena heurističkog pristupa u postupku pretrage puta do završnog čvora kod Dikstrinog algoritma smanjuje područje pretrage na manji broj čvorova što opisuje pristup po kojem A* algoritam funkcionira, te je njegova primjena kod autonomnih vozila prikazana na slici 7(b) [10].

Kod okoline u kojoj nisu definirana polja, točke ili čvorovi koristi se RRT algoritam koji samostalno generira svoje vlastite čvorove u matrici dozvoljenog prostora prikazanog na slici 7(c) . Čvorovi se stvaraju jedan za drugim, te je izlazna funkcija krivulja koja garantira kinematičku izvedivost. [4]

4.2 Umjetna inteligencija temeljena na odlukama

Umjetna inteligencija je sposobnost digitalnog računala ili računalno-kontroliranog robota da izvodi zadaće obično povezane uz inteligentna bića [11]. Primjena umjetne inteligencije (AI) u autonomnoj vožnji se ostvaruje preko njezine sposobnosti da simulira vozačeve kognitivne sposobnosti i učenje, te sudjeluju u funkcijama donošenja odluka. Primjena umjetne inteligencije se isto tako često koristi u mobilnoj robotici.

Umjetna inteligencija koja je temeljena na odlukama predstavlja kompleksne sustave koji rješavaju specifične zadatke na temelju baze podataka koja preko postavljenih pravila donosi određene odluke u vožnji. Prednosti ovog sustava su ti što njihova postava dobro simulira ljudsku logiku i racionalnost. [12]

Nedostatci koji se pojavljuju su kružno rasuđivanje i iscrpno dodavanje pravila koja dovode do beskonačnih petlji i utječe na vrijeme procesiranja. U slučaju da situacija u vožnji se ne poklapa sa bazom podataka pravila ili pravilnikom, onda se uzima standardna odluka koja možda neće biti prikladna za zadanu situaciju.

5 Sklopovska i programska podrška simulatora autonomnih vozila

Kako bi se objasnilo na koji način simulator autonomnog vozila funkcionira potrebno je objasniti programsku i sklopovsku podršku koja ga i pokreće. U sljedećim poglavljima će se objasniti glavni sustav (Arduino) simulatora autonomnog vozila, te njegovi senzori i princip rada.

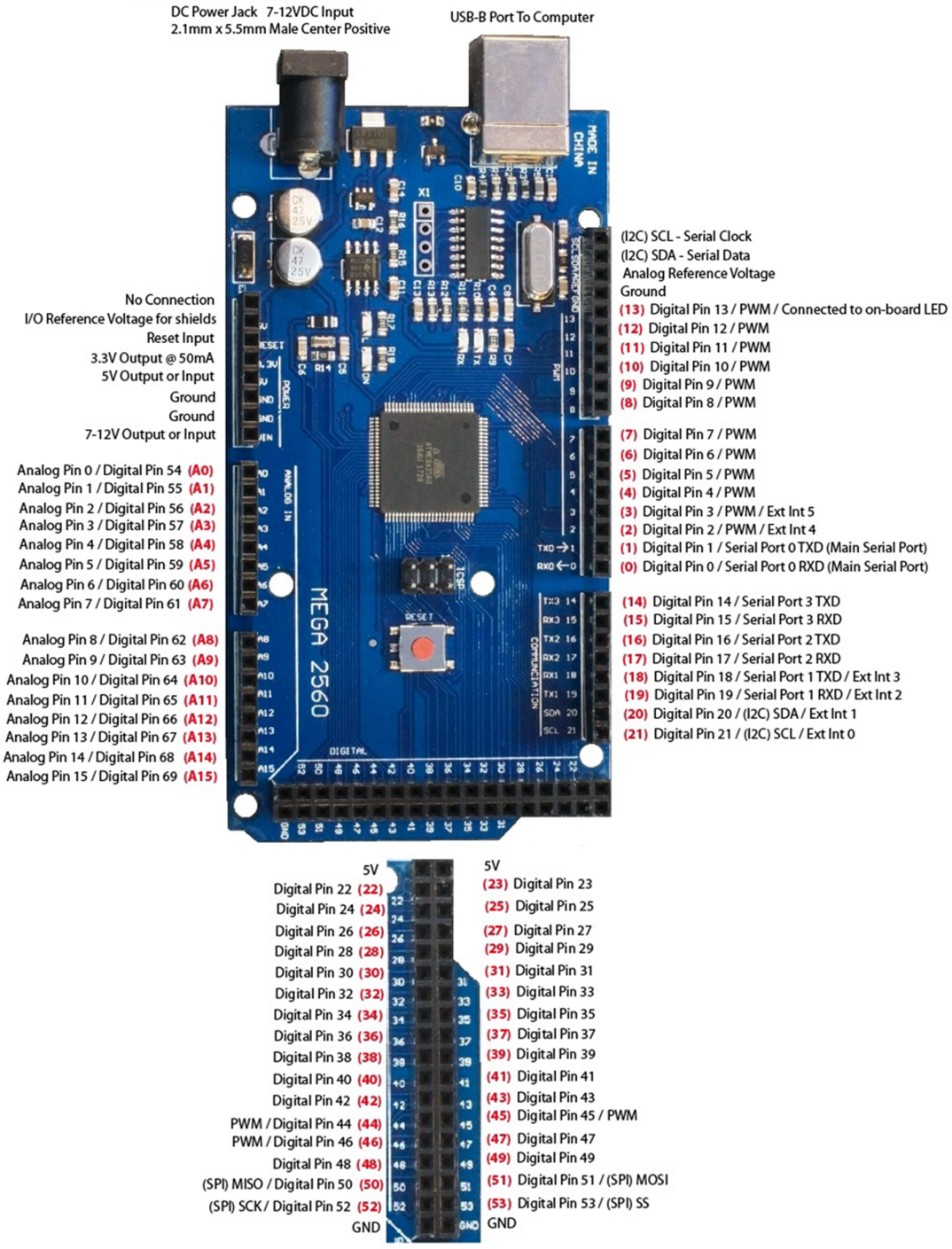
5.1 Razvojno okruženje Arduino

Arduino je otvorena platforma za razvoj različitih projekata iz područja robotike, prometa i slično. Ljudi koji tek počinju sa elektronikom često uzimaju Arduino kao svoju prvu razvojnu platformu. Arduino se sastoji od fizičke programibilne tiskane pločice (mikrokontroler) i programske podrške ili integriranog razvojnog okruženja (IDE, engl. Integrated Development Environment) koji radi na računalu i koristi se kako bi se pisao i prenosio računalni kod na fizičku ploču [13].

Arduino kao cjelina ne treba dodatnu sklopovsku podršku (programator) kako bi se prenio novi kod u mikrokontroler, nego se jednostavno koristi USB kabal. Arduino IDE koristi pojednostavljenu verziju C++-a čime pojednostavljuje sam proces učenja i izrade samog programa.

5.1.1 Arduino ATmega 2560

Arduino 2560 je tiskana pločica s mikrokontrolerom koja se zasniva na mikrokontroleru ATmega2560. Ima 54 digitalnih ulazno/izlaznih pinova od kojih se 15 može koristiti kao PWM (engl. Pulse-Width Modulation) izlaz, 16 analognih ulaza, 4 UART (sklopovka sučelja za serijsku komunikaciju), 16 MHz kristalni oscilator, USB priključak, utičnicu za napajanje, ICSP zaglavlje i tipku za resetiranje što je vidljivo na slici 8. Sadrži sve što je potrebno za podršku mikrokontrolera. Jednostavno se spaja na računalo pomoću USB kabela, AC-DC adapterom ili baterijom da bi se započeo rad. [14]



Slika 8: Tehničke karakteristike Arduino ATmega 2560 razvojne pločice

Izvodljivost Arduino razvojne pločice dosta ovisi i izvoru napajanja za kojeg je preporučljivo da bude između 6 i 12 V. Izvor napajanja može biti USB kabel koji se priključuje na samo računalo ili vanjski izvor napajanja kao na primjer baterija od 9V. Arduino MEGA koristi niz pinova koji se dijele na PWM (Power Module), ARF (Analog

Reference), naponske (3,3V; 5V; GND), te analogne i digitalne ulazno/izlazne pinove koji su objašnjeni u sljedećem:

1. 5V i 3,3V: Napajanje - naponski pinovi koji omogućuju napajanje različitim komponentama, te kao gornje granice imaju postavljene 5V i 3,3V;

2. GND (Ground) - Masa ili uzemljenje. Arduino MEGA ima 5 GND pinova od kojih možemo bilo koji iskoristiti kako bismo uzemljili krugove;

3. DIGITAL - digitalni ulazi (npr kod korištenja tipkala Arduinu se šalje stanje tipkala koje može biti 0 ili 1) i digitalni izlaz (npr. napajanje LED diode);

4. ANALOG IN: signali koji se šalju iz analognih senzora (npr. ultrazvučni senzor, senzor teperature itd.) zaprimaju se u analogne pinove, te se pretvaraju u digitalnu vrijednost koja se može pročitati;

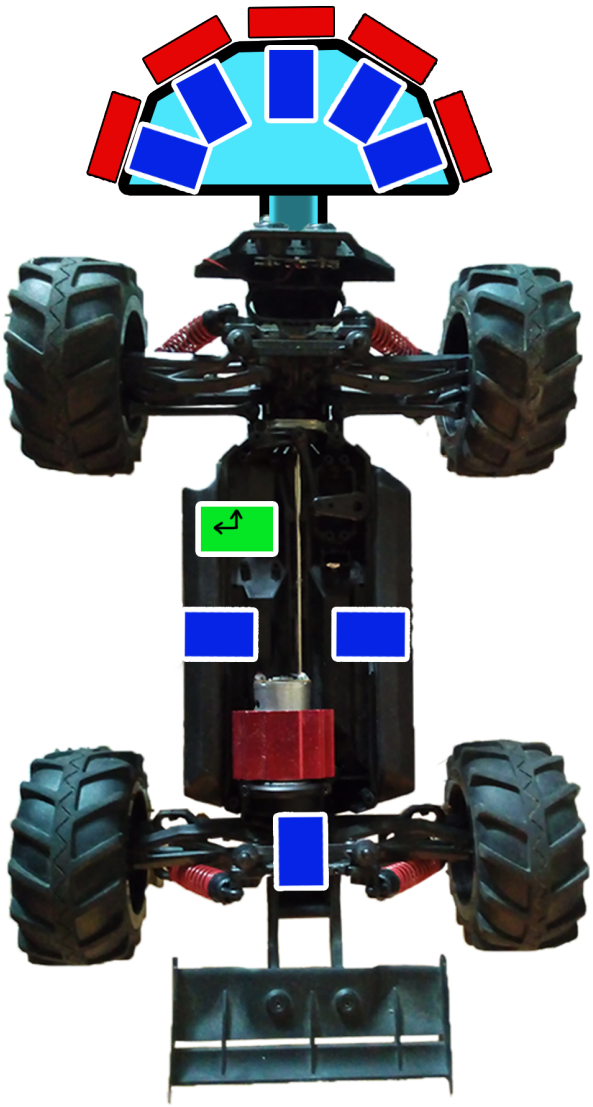
5. PWM - izlaz na kojem je moguć signal(napon) u rasponu od 0V do 5V zbog mogućnosti pina da modulira taj isti signal. Koristi se razne primjene gdje je potrebna dinamična promjena raspona signala na izlazu kao npr. upravljanje brzinom motora ili promjena jačine svijetljenja LED-diode;

6. AREF - pinovi koji se koriste kako bi se postavio vanjski referentni napon (od 0V do 5V) za analogne ulazne pinove.

5.2 Mjerne komponente simulatora autonomnog vozila

Jednostavnim programskim kodom Arduino može komunicirati s velikim brojem mjernih komponenti, te ih isto tako i kontrolirati. Postoje različiti tipovi mjernih komponenti ovisno o osjetilima koje koriste: osjetila temperature, udaljenosti, tlaka , ubrzanja, ugljičnog monoksida, svjetlosti, barometarskog tlaka i tako dalje.

Kako bi simulator autonomnog vozila prikupljao podatke iz njegove okoline, te ih onda proslijedio klijentskom sustavu što je u našem slučaju Arduino razvojna platforma potrebni su mu različiti senzori (ultrazvučni, infracrveni senzori i žiroskop) .



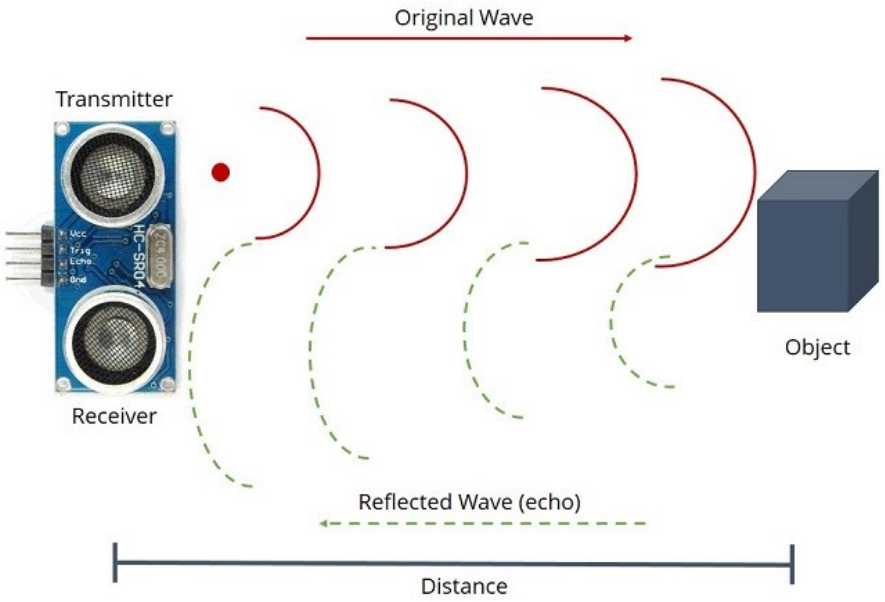
Slika 9: Postava senzora na maketi autonomnog vozila (ljubičasta - infracrveni senzori; crvena - ultrazvučni senzori; zelena - žiroskop)

Na slici 9 prikazana je postava svih mjernih komponenti na simulatoru autonomnog vozila, koji se sastoje od HC-SR04 ultrazvučnih senzora za srednji/dugi domet, KY-032 infracrvenih senzora za izbjegavanje prepreka za kratki domet, te ADXL345 žiroskopa i akcelerometra za detekciju nagiba koji će biti bolje objašnjeni u sljedećim pod-poglavljima.

5.2.1 Ultrazvučni senzori

Ultrazvučni senzor HC-SR04 je mjerna komponenta koja se često koristi u kombinaciji Arduino razvojnom platformom ili Raspberry PI mikro-računalom. HC-SR04 koristi sonar kako bi odredio udaljenost od nekog objekta na sličan način kako to čine i šišmiši po noći [15]. HC-SR04 koristi pinove kao što su GND (Masa-uzemljenje), VCC (napajanje), te ECHO (prijam signala) i TRIGG (odašiljanje signala).

Kako bi se generirao ultrazvuk treba se postaviti TRIGG pin na visoko stanje za 10 qs što će poslati kratak sonični signal u osam ciklusa prikazanom na slici 10 koji će se nakon odbijanja od objekta zaprimit na ECHO pinu. ECHO pin će na izlazu emitirat vrijeme u mikrosekundama koliko je trajalo putovanje zvučnog vala.



Slika 10: Princip na kojem funkcionira HC-SRO4 ultrazvučni senzor

Kako na ECHO pinu dobijamo dvostruku vrijednost udaljenosti jer se uzima u obzir i povratak signala prvo moramo izračunati koliko je zapravo trajalo putovanje u mikrosekundama, te nakon toga vrijeme pomnoženo sa brzinom gibanja zvučnog vala podijeliti sa dva.

Trajanje putovanja signala se računa prema priloženoj formuli:

$$t = \frac{s}{v} \tag{1}$$

Gdje je:

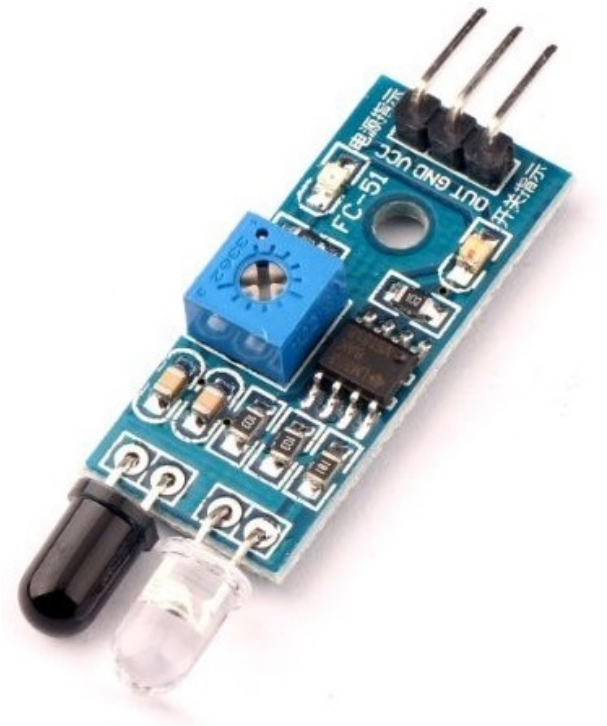
t - vrijeme putovanja zvučnog vala;

s - udaljenost objekata u cm;

v - brzina svjetlosti (0.034 cm/qs).

5.2.2 Infracrveni senzori za izbjegavanje objekata

Infracrveni senzor za izbjegavanje objekata KY-032 je mjerna komponenta koja vraća signal kada detektira bilo kakvu prepreku u zadanom perimetru koji se može kod većine ovakvih senzora namještati preko potenciometra. Perimetar detekcije varira ovisno o postavka potenciometra između 2 do 40 cm. Radni napon iznosi između 3,5 do 5V dok je struja 20 mA. [16]

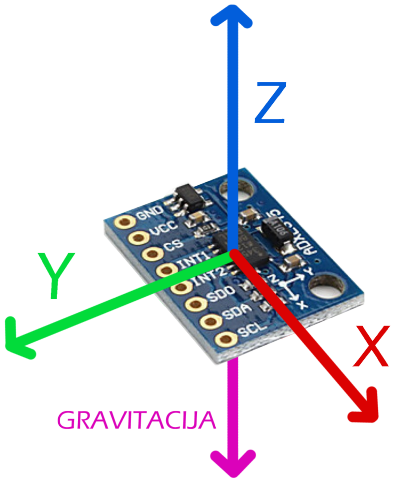


Slika 11: Infracrveni senzor za izbjegavanje objekata KY-032

Uređaj prikazan na slici 11 ima infracrveni odašiljač i prijamnik koji tvore senzorski par gdje LED-dioda odašilje određenu frekvenciju infracrvenog signala kojeg druga LED-dioda detektira. LED-prijamnik kada primi signal će okinuti digitalno stanje koje može biti 0 ili 1 ako objekt uđe u određeni perimetar ispred senzora.

5.2.3 Žiroskop

Žiroskop ADXL-345 je akcelerometar koji može mjeriti statičnu i dinamični silu akceleracije u tri osi (x,y,z) prikazane na slici 12 [15]. Zemljina gravitacijska sila je jedan od primjera statične sile koja se mjeri, dok dinamične sile su izazvane preko pokreta, vibracija i slično.



Slika 12: Postava x,y i z osi kod ADXL345 žiroskopa i akcelerometra

Kako bi dobili kut nagiba u x (Roll) i y (Pitch) osi korištene su sljedeće formule:

$$x_{nagib} = arctan \cdot \frac{-Gx}{Gz} \tag{2}$$

$$y_{nagib} = arctan \cdot \frac{Gy}{\sqrt{Gx^2 + Gz^2}} \tag{3}$$

Gdje je:

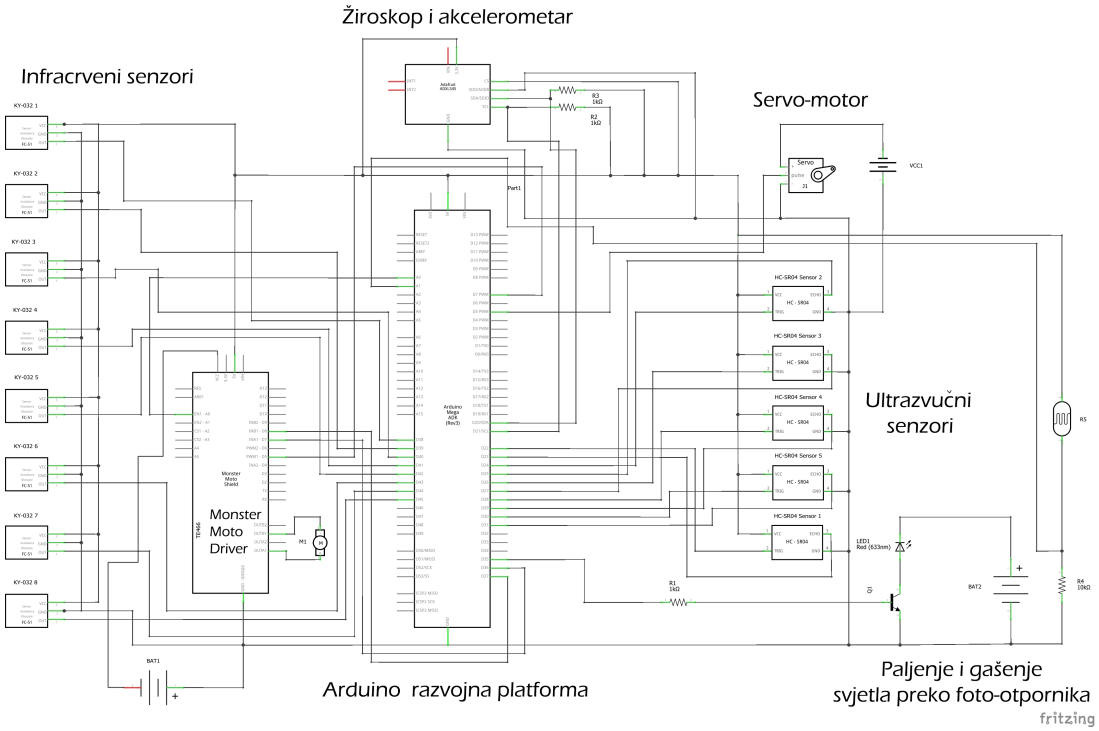
Gx - akceleracija u x smjeru;

Gy - akceleracija u y smjeru;

Gz - akceleracija u z smjeru.

5.3 Sklop simulatora autonomnog vozila

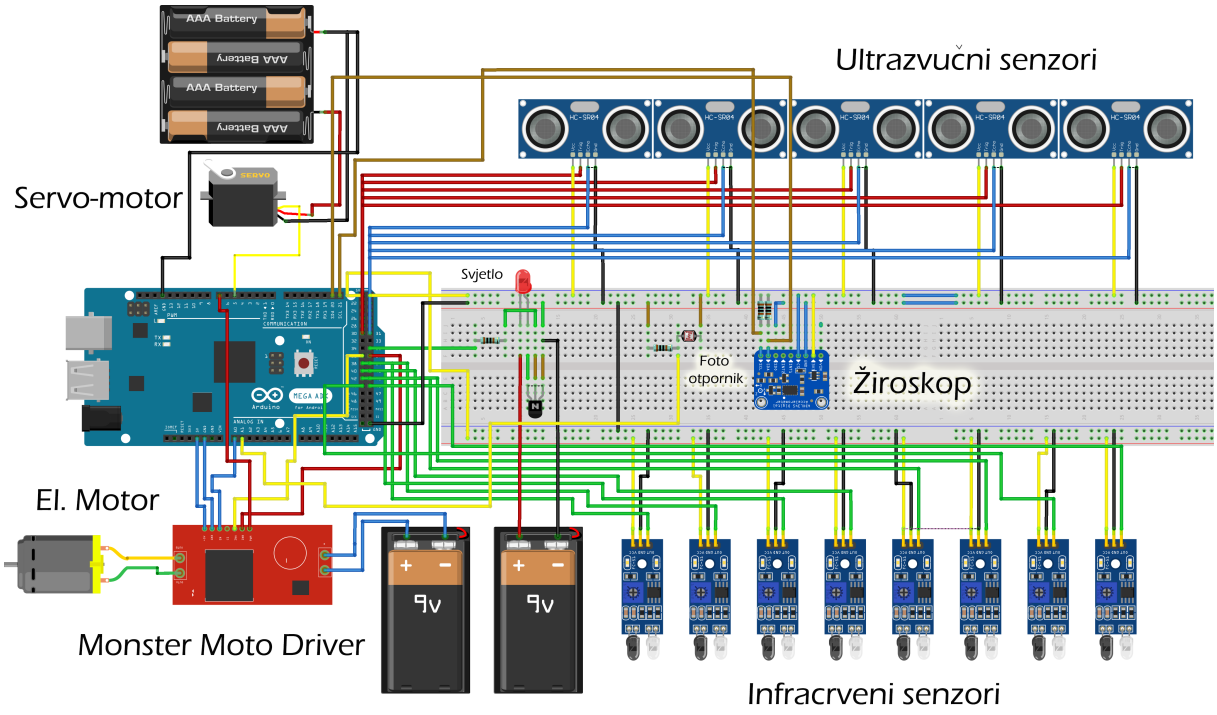
Sklop simulatora autonomnog vozila sastavljen je od Arduino razvojne platforme, makete malog cestovnog vozila, tranzistora, baterije (7,2V), infracrvenih senzora, ultrazvučnih senzora, žiroskopa, Monster Moto Driver za kontrolu motora velike snage, te foto-otpornika. Nakon izrade samog sklopa i učitavanja programskog koda za njegovo upravljanje u Arduino razvojnu pločicu, započinje njegovo izvršavanje koje se sastoji od kinematičkog kretanja, te izabiranja najveće udaljenosti preko ultrazvučnih senzora za kretanje naprijed gdje se uz to provjeravaju stanja na infracrvenim senzorima zbog mogućnosti nailaska na bliske objekte koje ultrazvučni senzori nisu detektirali. Slika 13 prikazuje shemu spajanja mjernih i pogonskih komponenti sa Arduino razvojnom pločicom, te tranzistora kao sklopku za paljenje u gašenje svjetla na vozilu u slučaju zamračenja okoline.



Slika 13: Elektronička shema sklopa simulatora autonomnog vozila

Kako bi se bolje predočio sklop samog upravljačkog sustava simulatora autonomnog vozila potrebno je izraditi maketu komponenti prikazanu na slici 14 koja se sastoji od upravljačkih grupa kao što su grupa infracrvenih senzora, ultrazvučnih senzora, Arduino

razvojne pločice kao glavne procesne jedinice, žiroskopa i grupe za detekciju zamračenja okoline preko foto-otpornika. Vizualna percepcija makete komponenti igra važnu ulogu u olakšavanju spajanja kompleksnijih sustava, te i ubrzava sam proces planiranja i izrade.

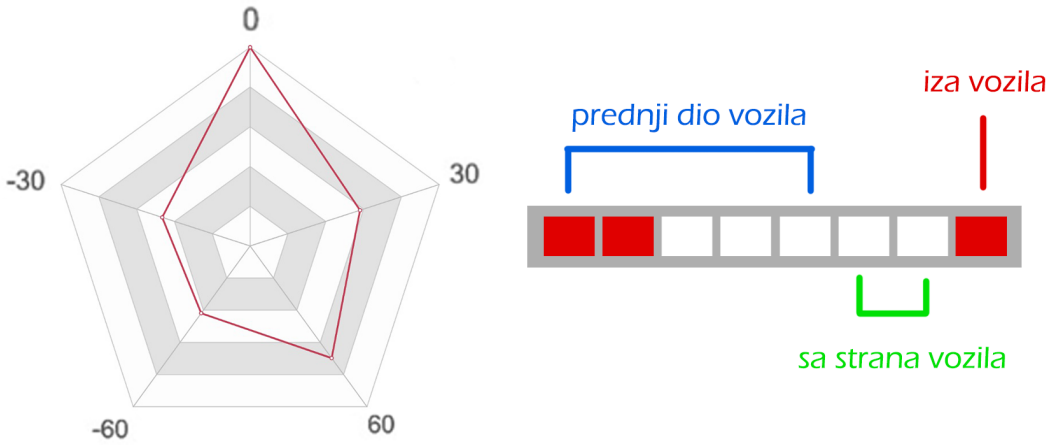


Slika 14: Eksperimentalna maketa komponenti simulatora autonomnog vozila

5.3.1 Princip rada simulatora autonomnog vozila

Simulatorom autonomnog vozila upravlja programski kod koji simulira upravljački sustav primajući konstantne informacije iz senzora postavljenih na maketi autonomnog vozila, te djeluje ovisno o tim informacijama. Kompletan kod je zasnovan na obradi informacija zaprimljenih iz senzora, pretvaranju tih informacija u razumljive podatke, te jedno-iteracijskoj pohrani tih informacija u memoriju kako bi se kasnije tijekom samog izvođenja kinematičkih manevara mogle koristiti za usmjeravanje. Kada se elementi simulatora autonomnog vozila kao što su servo-motor namijenjen za skretanje i Arduino razvojna platforma spoje na odgovarajući izvor napajanja dolazi do postavljanja senzora u stanje očitanja. Ultrazvučni senzori se prvo moraju "naviknuti" na prostor oko sebe, te time njihove prvotne vrijednosti nisu u potpunosti točne zbog čega ih je potrebno odbaciti prije nego se učitaju u privremenu bazu podataka kao polje. Infracrveni senzori

za izbjegavanje objekata ne trebaju se priviknut na okolinu te oni automatski započinju očitavanje u slučaju da se početna pozicija vozila nalazi neposredno uz neki objekt kako bi se odmah započeo manevar izbjegavanja. Na slici 15 je prikazana vizualna predodžba podataka koje simulator autonomnog vozila prosljeđuje dalje na obradu, te je to isto i način na koji vrši percepciju okoline oko sebe.



Slika 15: Vizualna percepcija podataka ultrazvučnih(lijevo) i infracrvenih senzora(desno)

Upravljački sustav postavlja standardnu brzinu, te žiroskop očitavajući nagibe u stvarnom vremenu pojačava u slučaju inklinacije ili smanjuje snagu motora u slučaju deklinacije nagiba u y osi. Nakon očitanja i postave očitanih podataka u matrice koje služe kao privremeni spremnik za jednu iteraciju glavne petlje programskog koda šalje se naredba za pokretanje motora kako bi se vozilo počelo kretati, te nakon toga se upućuje servo-motoru koji i upravlja skretanjem pod kojim kutem da postavi kotače. Kotači se pri vožnji mogu postaviti na pet različitih pozicija. Pozicije 45°, 22.5° se koriste kod skretanja u desno, dok -45°, -22.5° se koriste kod skretanja u lijevo, te ima standardna pozicija koja je 0° i koristi se za pravo.

Prije samog procesa skretanja i drugih različitih manevara potrebno je prikupljene podatke procesirat i prema postavljenoj logici u programu odlučiti o sljedećim radnjama u vožnji. Glavni princip je da se vozilo treba uputiti prema najvećoj očitanoj udaljenosti i u slučaju da je najveća udaljenost očitana sa ultrazvučnog senzora koji je pod kutem, onda se šalje naredba servo motoru da zakrene kotače prema toj strani na određeni broj sekundi,

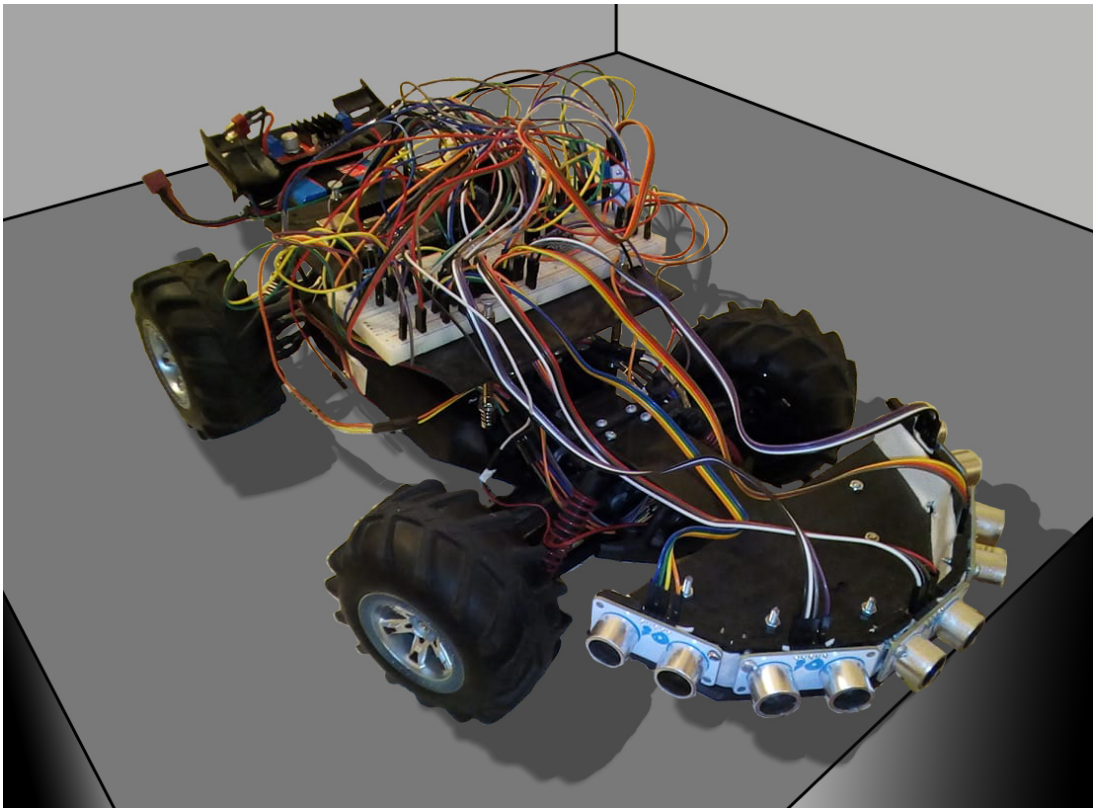
te ih vrati u prvotnu poziciju. Na primjer ako je najveća očitana udaljenost bila 83 cm očitana sa ultrazvučnog senzora postavljenim pod kutem od 60° onda će servo postaviti poziciju kotača na 45° za onoliko sekundi koliko je potrebno dok se ponovo ne očita u svim sljedećim iteracijama glavne petlje najveća udaljenost na senzoru postavljenim pod kutom od 0°. Kako ultrazvučni senzori ponekada ne vrate ispravnu vrijednost udaljenosti dolazi do neposredne opasnosti od sudara, te u tom slučaju infracrveni senzori detektiraju objekt i postavljaju elektromotor u stanje za kretanje unatrag.

Pri kretanju unatrag važno je uzeti u obzir dvije stvari, a to su udaljenosti objekata pored vozila i logičko stanje infracrvenog senzora postavljenog na kraj vozila. Vozilo će se kretati unatrag za onoliko sekundi koliko je definirano u programskom kodu u slučaju da infracrveni senzor na kraju vozila ne detektira prepreku, u suprotnom zaustavlja se kretanje unatrag i ponovo započinje kontinuirano kretanje unaprijed. Ako je udaljenost objekata pored vozila u nedozvoljenom perimetru koji iznosi od 10 do 15 cm pri kretanju unatrag, servo motor zakreće kotače prema toj strani za 45° na kojoj se nalazi objekt i time mijenja cjelokupno usmjerenje vozila.

5.3.2 Izrada makete simulatora autonomnog vozila

Maketa simulatora autonomnog vozila sastoji se od postojeće makete malog cestovnog vozila, Arduino programatora, steznika za ventilatore procesora stolnog računala iskorištene kao držač za postolje, plastičnog postolja za Arduino razvojnu pločicu i eksperimentalnu pločicu koja je poslužila za olakšavanje procesa grananja vodiča, plastičnog postolja za postavu mjernih komponenti pod određenim kutem, vodiča, te ostalih mjernih (ultrazvučni, infracrveni senzori i žiroskop) i pogonskih komponenti (elektromotor, upravljač elektromotornog pogona VNH2SP30 Monster Moto Driver i servo motor). Proces izrade je podijeljen u više faza izrade. Izrada je krenula od planiranja upravljačkog sustava simulatora i odabira odgovarajućih komponenti, a tek način i gdje će te iste komponente biti instalirane na postojećoj maketi malog cestovnog vozila. Nakon toga kreće spajanje steznika za ventilatore procesora na podnožje postojeće makete i instalacija plastičnog postolja na njega. U sljedećoj fazi se na prednji dio makete malog cestovnog vozila instaliralo plastično postolje za postavu mjernih komponenti pod određenim kutem, te postavljanja istih komponenti na to postolje. U zadnjoj fazi smo postavili pogonske komponente na

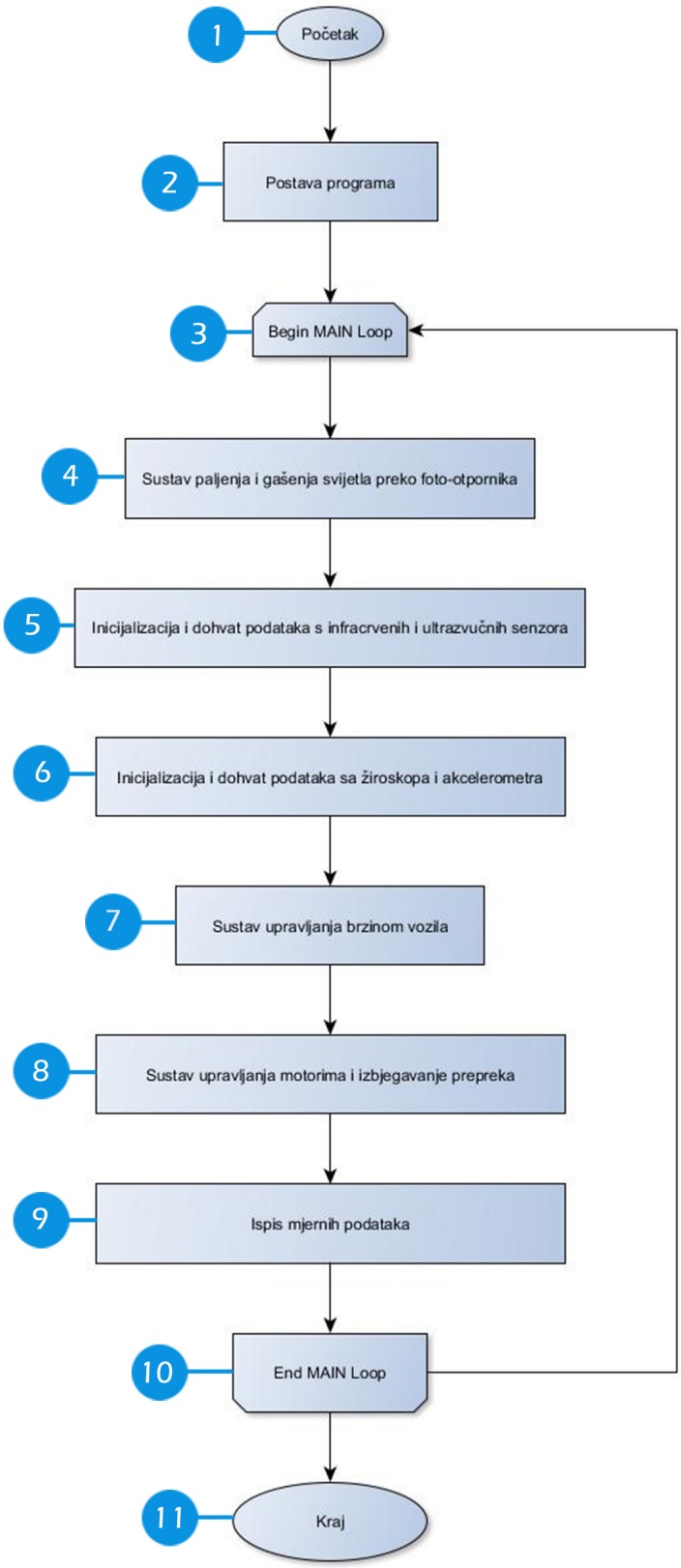
spojler makete vozila i povezali elektromotor sa mehaničkim sustavima koji prenose okretni moment elektromotora na kotače makete. Kako bi se uopće moglo započeti testiranje potrebno je povezati sve odgovarajuće komponente s Arduino programatorom i učitati programski kod u njegovu memoriju (Slika 16).



Slika 16: Maketa simulatora autonomnog vozila

5.4 Program za upravljanje simulatorom autonomnog vozila

U ovom poglavlju će biti prikazana logika rada napravljene programske podrške kroz prikaz pojednostavljenog dijagrama toka(Slika 17) i programskog koda priloženog u prilogu 1. Programski kod dostupan je na stranici: `https://github.com/Kiki6667/Pregled_Upravljackih_Sustava_Autonomnih_Vozila.git`



Slika 17: Pojednostavljeni dijagram toka rada simulatora autonomnog vozila

U nastavku je dan opis izvršavanja dijagrama toka prema blokovima označenim u dijagramu toka prikazanom na slici 17 .

1. Uključivanje napajanja.

2. Postavljanje programa se sastoji od deklaracija potrebnih biblioteka za rad određenih komponenti ili za korištenje matematičkih funkcija. Isto tako je potrebno deklarirati potrebne varijable koje će se koristiti za obradu podataka ili privremeno spremanje podataka u memoriju mikrokontrolera. Kako bi kasnije tijekom izvršavanja glavne programske petlje mogli pozivati odgovarajuće funkcije za obradu podataka ili kontrolu komponenti pogonskog sustava potrebno ih je definirati na početku.

3. Započinje izvršavanje glavne programske petlje koja će se beskonačno ponavljati sve dok je Arduino razvojna pločica spojena na napajanje ili ako ju ne prekinemo sa internom naredbom.

4. Sa analognog ulaza na Arduino razvojnoj pločici se u stvarnom vremenu očitava otpor foto-otpornika. U slučaju da dođe do pada intenziteta svjetlosti u okolini otpor će porasti iznad $600\,\Omega$ čime se šalje naredba digitalnom izlazu da okine tranzistor koji će kao sklopka spojiti ili odspojiti napajanje (7V) na svjetla simulatora autonomnog vozila.

5. Inicijaliziraju se funkcije za dohvat podataka očitanja infracrvenih i ultrazvučnih senzora definirane u drugom bloku programskog koda, te se podaci spremaju u zasebna polja koja će biti pohranjena u memoriju mikrokontrolera samo za vrijeme trajanja jedne iteracije.

6. ADXL345 žiroskop i akcelerometar se postavlja u stanje očitanja, te se čitaju akceleracije u x, y, z osi. Nakon očitanja se prema formulama 2 i 3 računa nagib u x i y osi.

7. Inicijalizira se funkcija za odabir odgovarajuće brzine kojoj se šalje podatak nagiba u y-osi kako bi vozilo povećalo brzinu u slučaju pozitivnog nagiba (inklinacije) ili smanjilo brzinu u slučaju negativnog nagiba(deklinacije).

8. Učitavaju se polja memoriranih podataka ultrazvučnih i infracrvenih senzora, te se prema principu objašnjenom u poglavlju 5.3.1 određuje usmjerenje, paljenje gašenje motora, smjer motora, kutevi postavljanja prednjih kotača, kretanje unatrag itd.

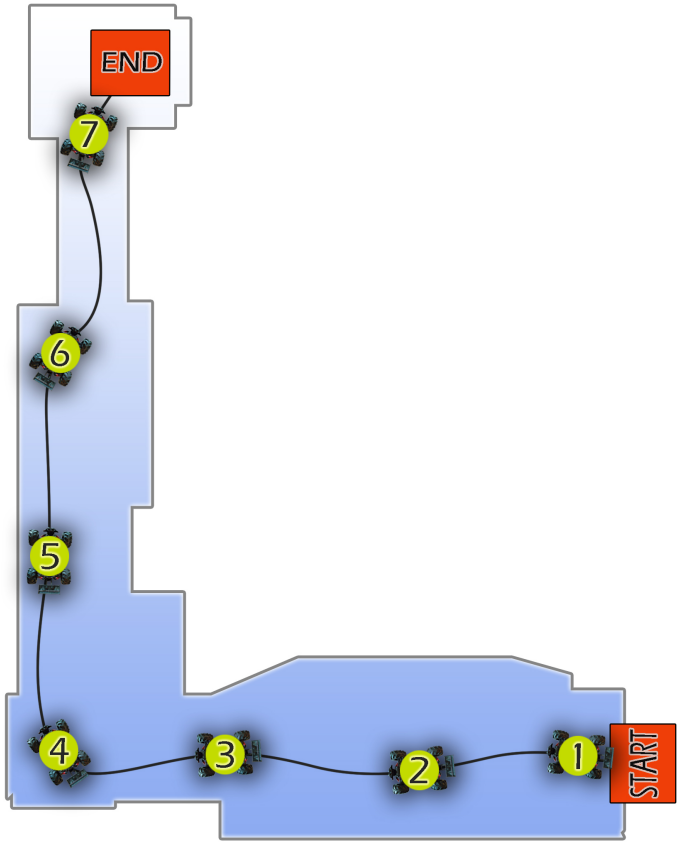
9. Kako bi se prilikom testiranja autonomnog vozila prikupljeni podaci iz senzora mogli pročitat i memorirat potrebno ih je ispisat u serijskom sučelju programske podrške Arduino razvojne platforme. Ispisuju se udaljenosti svih ultrazvučnih senzora, stanja infracrvenih senzora, nagibi očitani od strane žiroskopa u x, y, z osi, te samo vrijeme očitanja.

10. U slučaju okidanja interne naredbe za prekid petlje zbog određenog uvjeta, program će prekinut izvršavanje glavne programske petlje, u suprotnom ciklus se nastavlja.

11. Program završava isključivanjem napajanja ili okidanjem interne naredbe za prekid glavne petlje.

6 Testiranje simulatora autonomnog vozila

Kako je razina pouzdanosti veoma važan atribut kod razlike između autonomnih vozila 4 i 5 stupnja prema SAE klasifikaciji, potrebno je izvršiti različite testove koji bi utvrdili tu istu razinu pouzdanosti kod potencijalnog autonomnog vozila. Tako je potrebno u ovom slučaju napraviti mali test koji se sastoji od poligona prikazanog na slici 18 po kojem će se simulator autonomnog vozila kretati, prijenosnog računala za praćenje i memoriranje podataka spojenog na simulator autonomnog vozila i samog simulatora autonomnog vozila kako bi se pokazalo koliko je zapravo sustav simulatora pouzdan kod izbjegavanja prepreka ili pronalaska optimalnog puta.



Slika 18: Putanja izbjegavanja simulatora autonomnog vozila na testnom poligonu

Pri testiranju u programskom kodu je postavljena standardna brzina koja se ne mijenja pri kinematičkim manevrima iz razloga praktičnosti pri memoriranju podataka na

prijenosno računalo. Tijekom cijelog testa prijenosno računalo je preko USB-kabela moralo biti spojeno na simulator kako bi simulator ispisivanjem podataka udaljenosti očitanih od strane ultrazvučnih senzora ili stanja infracrvenih senzora u stvarnom vremenu mogao memorirat iste u serijskom sučelju programske podrške Arduino razvojne platforme. Pri postavljanju objekta na poziciji 2 na slici 18, simulator autonomnog vozila je uspješno izbjegao navedeno, no pri prelasku u poziciju 3 u više navrata su se loše očitale udaljenosti gdje je ultrazvučni senzor na 60° uvijek očitao najveću udaljenost i time konstantno usmjeravao simulator autonomnog vozila u desno što je rezultiralo beskonačnoj kružnoj putanji između pozicije 3 i 1. Pri kasnijim testiranjima objekt je bio maknut i test se izvodio bez njega. Pri dolasku u poziciju 4 ultrazvučno senzori su očitali najveću udaljenost sa desne strane i time uspješno usmjerile vozilo, koje se sekundu kasnije zaustavilo zbog detekcije prepreke od strane infracrvenih senzora i započelo kretanje unatrag kako bi se ispravilo u sukladnosti prema najvećoj očitanoj udaljenosti. Nakon što je simulator autonomnog vozila uspješno odradio cijelu određenu rutu kroz pozicije 5,6 i 7 prema logici opisanoj u poglavlju 5.3.1 dobijeni su izlazni podaci u serijskom sučelju priloženi u tablici 1.

Sakupljeno je 93 zapisa u toku cijelog testa od početka do kraja predviđene rute, te su izdvojeni najvažniji koji se poklapaju sa položajima navedenih pozicija na tlocrtu testnog poligona. Podaci S1 do S5 predstavljaju udaljenosti koje su vratili ultrazvučni senzori, te koji pokazuju prema najvećoj udaljenosti na koju stranu će se vozilo usmjeriti. X-gyro, Y-gyro, te Z-gyro predstavljaju nagibe žiroskopa po pojedinoj osi, te su priloženi više iz razloga da se prikaže kao primjer. U slučaju da jesu postojali nagibi Y-gyro varijabla bi mijenjala vrijednost time bi se prosljeđivala sustavu za kontrolu brzine koji bi pri inklinaciji povećao brzinu. Polje infracrvenih senzora Infra_sen se sastoji od osam stanja (slika 15) u kojem prvih pet s lijeva predstavljaju infracrvene senzore postavljene ispred vozila i na istim kutovima kao što su pozicije ultrazvučnih senzora, druga dva stanja infracrvene senzore postavljenje sa strana vozila, te zadnje stanje koje predstavlja senzor iza vozila koji se aktivira ako pri kretanju unatrag vozilo naiđe na prereku što prekida trenutni kinematički manevar i ide na sljedeći. Infracrveni senzori su se tek počeli aktivirati od pozicije 4 do 7 jer se simulator autonomnog vozila kretao jako blizu zidu kao prepreki, te pri samom kraju gdje se aktivirao infracrveni senzor postavljen na 0° ili S3 što je

označavalo prepreku ispred samog vozila i time je testiranje bilo završeno.

Tablica 1: Izlazni podaci testa izbjegavanja prepreka simulatora autonomnog vozila

Pozicija Vrijeme [ms]	**S1**	**S2**	**S3**	**S4**	**S5**	**X-gyro**	**Y-gyro**	**Z-gyro**	**Infra_sen**
1 1606	28	100	132	68	31	0.11	0.02	0.96	00000000
2 11242	85	92	175	102	90	0.16	−0.04	0.81	00000000
3 23287	34	90	110	115	40	0.03	0.04	0.82	00000000
4 35332	15	19	30	130	99	0.21	−0.02	0.95	01000000
5 47377	5	8	90	68	43	0.26	−0.09	0.83	11000000
6 59422	7	64	76	56	63	−0.08	−0.14	1.09	01000000
7 73073	22	13	15	22	23	0.14	−0.02	0.98	10100000

7 Zaključak

Od svih vrsta prometa automobilski promet ima najveći postotak smrtnih slučajeva. Primjenom autonomnih vozila došlo bi do smanjenja smrtnih slučajeva uzrokovanih stanjem umora ili iscrpljenosti, pijanim stanjem i slično da je pouzdanost u sustave autonomnih vozila neupitna.Kako je nepouzdanost i preduvjet za sva vozila koja su u potpunosti autonomna ili djelomično autonomna potpuno pouzdanje u sustave je teško postići zbog dinamične i kompleksne okoline u kojima mnogi faktori igraju ulogu.

U ovom radu obrađeni su standardni upravljački sustavi autonomnog vozila gdje je obrađena povijest istog kako bi se dobio uvid u sam razvitak upravljačkih sustava, te su zasebno opisano elementi upravljačkog sustava kao što su senzorski, klijentski, akcijski i korisnički sustavi. Opisane su različite pogonske i mjerne komponente simulatora autonomnog vozila koji predstavljaju akcijske i senzorske sustave standardnog autonomnog vozila i sam princip rada simulatora autonomnog vozila. Primijenjena je Arduino razvojna platforma s kojom se implementirao jednostavni sustav autonomnog vozila u maketu malog cestovnog vozila kako bi se i testirala na malom poligonu ta ista pouzdanost da će simulator autonomnog vozila doći od svoje startne do krajnje pozicije.

Arduino razvojna platforma je bila idealna za korištenje kao procesna jedinica simulatora autonomnog vozila ponajviše zbog svojih tehničkih karakteristika. Sa određenim testom na malom poligonu pokazalo se da različiti dijelovi upravljačkog sustava kao akcijski i klijentski sustavi, što su na ovom slučaju pogonske komponente i Arduino razvojna platforma mogu simulirat jednostavan sustav autonomnog vozila, ako zaprime odgovarajuće podatke od strane senzorskih sustava.

Moguće nadogradnje postojećeg upravljačkog sustava sustava bile bi izrada PCB pločice koja bi smanjila broj žica iskorištenih tijekom samog spajanja i time smanjila težinu cjelokupnog simulatora što je izrazito ključna stvar kod izvođenja različitih kinematičkih manevara. Druga nadogradnja bi bila puno složeniji algoritam koji bi upravljao istim, te isto tako instalacija ESP8266 mikrokontrolera koji poslužio za komunikaciju Arduino

razvojne platforme sa web serverom gdje bi se u stvarnom vremenu ispisivali podaci pročitano od strane različitih senzora.

Literatura

[1] Fabian Kröger. *Automated Driving in Its Social, Historical and Cultural Contexts*, pages 41–68. 05 2016.

[2] James M. Anderson, Nidhi Kalra, Karlyn D. Stanley, Paul Sorensen, Constantine Samaras, and Oluwatobi A. Oluwatola. *Autonomous Vehicle Technology: A Guide for Policymakers*. RAND Corporation, 2014.

[3] https://en.wikipedia.org/wiki/Waymo; pristupljeno: 10.08.2019.

[4] Laurene Claussmann, Marc Revilloud, Dominique Gruyer, and Sebastien Glaser. A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–23, 2019.

[5] Margarita Martínez-Díaz, Francesc Soriguera, and Ignacio Pérez. Autonomous driving: a birds eye view. *IET Intelligent Transport Systems*, 13(4):563–579, April 2019.

[6] Krajnović B. Navigacija autonomnih vozila. Završni rad, Sveučilište u Zagrebu, Fakultet prometnih znanosti, Zagreb, 2017. Dostupno na: https://urn.nsk.hr/urn:nbn:hr:119:523888.

[7] Hrga M. Računalni vid. Zbornik radova Veleučilišta u Šibeniku, vol., br. 1-2/2018, str. 207-216, 2018.[Online]. Dostupno na: https://hrcak.srce.hr/198597. [Citirano: 09.08.2019.].

[8] Rasheed Hussain and Sherali Zeadally. Autonomous cars: Research results, issues, and future challenges. *IEEE Communications Surveys & Tutorials*, 21(2):1275–1313, 2019.

[9] https://moodle.srce.hr/2018-2019/pluginfile.php/2265778/mod_resource/content/1/05-Satelitski_pozicijski_sustavi.pdf; pristupljeno: 09.08.2019.

[10] S. Liu, L. Li, J. Tang, S. Wu, and J. Gaudiot. *Creating Autonomous Vehicle Systems*. Morgan Claypool, 2017.

[11] L. Kovač, "Umjetna inteligencija danas", Diplomski rad, Sveučilište u Rijeci, Filozofski fakultet, Rijeka, 2015. Dostupno na: https://urn.nsk.hr/urn:nbn:hr:186:606497.

[12] Karsten Berns and Ewald von Puttkamer. *Autonomous Land Vehicles*. ViewegTeubner, 2009.

[13] Marko Slavulj. Mogućnosti primjene arduino razvojne platforme za simulaciju rada raskrižja. *URN*, Sep 2018.

[14] https://store.arduino.cc/usa/arduino-mega-2560-rev3, 15.08.2018.

[15] Jeremy Blum. *Exploring Arduino: Tools and Techniques for Engineering Wizardry*. John Wiley & Sons, Inc., USA, 1st edition, 2013.

[16] https://tkkrlab.nl/wiki/Arduino_KY-032_Obstacle_avoidance_sensor_module; pristupljeno: 13.08.2019.

Popis slika

Popis tablica

Prilog 1. Programski kod simulatora autonomnog vozila

```
#include<Servo.h>
#include<math.h>
//////////////////////////////////
const int analogIn = A7;
int mVperAmp = 100; // iskoristiti 100 za 20A Modul i 66 za 30A Modul
int RawValue= 0;
int ACSoffset = 2500;
double Voltage = 0;
double Amps = 0;
//////////////////////////////
unsigned long time=0;
float time_float=0;
//deklaracija trigger i echo pinova za senzore udaljenosti//
const int s1_trigg=24,s2_trigg=26,s3_trigg=28,s4_trigg=30,s5_trigg=32;
const int s1_echo=25,s2_echo=27,s3_echo=29,s4_echo=31,s5_echo=33;
////////////////////////////////////////
boolean x=false;
const int lamp = 37;
//////////////////////////////////////////////
///deklaracija varijabli za VNH2SP30 monster motor driver
#define MOTOR_A1_PIN 34
#define MOTOR_B1_PIN 35
#define PWM_MOTOR_1 6
const float I_current_value=A2;;
#define EN_PIN_1 A0
float ampers_motor;
////////////////////////////////////
int IRsensorARRAY[8]={};
//deklaracije za mjerenje struje
int Vout=A7;
float poljeStruja[10];
```

```
32 ////////////////////////////////
33 //polja varijabla udaljenosti i servo deklaracija
34 long duration;
35 int j=0;
36 int distance;
37 int distance_array[5]={};
38 int i=0;
39 Servo sv;
40 int rate=100;
41 int zbroj[5]={};
42 ////////////////////////////////////
43 //deklaracija varijabli za infracrvene senzore
44 const int IRout1=40,IRout2=41,IRout3=45,IRout4=42,IRout5=43;//
     deklaracija za prednje IR senzore
45 const int IRout_left=46,IRout_right=47,IRout_back=44;
46 //postava jedinicne matrice i polja stanja
47 int jed_matrica[3]={};
48 int polje_stanja[4][3]={};
49 ////////////////////////////////////////////
50 //za giroskop i akcelerometar///
51 #include <Wire.h>  // Wire library - koristi se za I2C komunikaciju
     izme u razlicitih mikrokontrolera
52 //u ovoom slucaju ziroskop
53 int ADXL345 = 0x53; //  I2C adresa ADXL345 senzora
54 float X_out, Y_out, Z_out;  // deklaracija varijabli za izlaze iz
     ziroskopa
55 float roll,pitch,rollF,pitchF=0;
56 //funkcije upravljanja koji se odnosi samo na motor//
57 void forward()
58 {
59         digitalWrite(MOTOR_A1_PIN, LOW);
60         digitalWrite(MOTOR_B1_PIN, HIGH);
61 }
62 void backward()
63 {
64   if(digitalRead(IRout_back) == HIGH)//ako nema objekata iza vozila onda
       onda idi unazad za svaki slucaj
65   {
66     digitalWrite(MOTOR_A1_PIN, HIGH);
```

```
67      digitalWrite(MOTOR_B1_PIN, LOW);
68    }
69  }
70  void brake()
71  {
72      digitalWrite(MOTOR_A1_PIN, LOW);
73      digitalWrite(MOTOR_B1_PIN, LOW);
74  }
75  ////////////////////////////////////////
76  void manevar_okretanja_45()
77  {
78    backward();
79    delay(800);
80    brake();
81    delay(500);
82  }
83  //dio upravljanja koji se odnosi samo na servo motor///
84  void straight()
85  {
86  sv.write(105);
87  }
88  void left_30()
89  {
90      sv.write(65);
91  }
92  void right_30()
93  {
94    sv.write(170);
95  }
96  void right_15()
97  {
98    sv.write(137);
99  }
100 void left_15()
101 {
102     sv.write(85);
103 }
104 ////////////////////////////////////////////////////
105 int odabir_smjera(int udaljenosti[5])
```

```
{
  int i=0;
  int zam=udaljenosti[0];
  for(i=1;i<=4;i++){
    if(udaljenosti[i]>zam)
    {
      zam=udaljenosti[i];
    }
  }
    if(zam==udaljenosti[0])
  {
    left_30();
    millis();
    return 1;
  }
  else if(zam==udaljenosti[1])
  {
    left_15();
    millis();
    return 2;
  }
  else if(zam==udaljenosti[2])
  {
    straight();
    return 3;
  }
 else  if(zam==udaljenosti[3])
  {
    right_15();
    millis();
    return 4;
  }
  else if(zam==udaljenosti[4])
  {
    right_30();
    millis();
    return 5;
  }
}
```

```
145 void speed_control(float Y_os)
146 {
147 int i=0;
148 if(Y_os<=0.1&&Y_os>0)
149 {
150   analogWrite(PWM_MOTOR_1,34);
151 }
152 else if(Y_os<=0.4&&Y_os>-(0.2))
153 {
154   analogWrite(PWM_MOTOR_1,35);
155 }
156 else if(Y_os<=0.6&&Y_os>0.4)
157 {
158   analogWrite(PWM_MOTOR_1,36);
159 }
160 else if(Y_os<=0.75&&Y_os>0.6)
161 {
162   analogWrite(PWM_MOTOR_1,40);
163 }
164 else if(Y_os<=1.3&&Y_os>0.75)
165 {
166   analogWrite(PWM_MOTOR_1,45);
167 }
168 else
169 {
170   analogWrite(PWM_MOTOR_1,34);
171 }
172 }
173 //funkcija za izracunavanje udaljenosti sa svakog ultrazvucnog senzora
174 int distance_calc(int a)
175 {
176   switch(a)
177   {
178   case 0:
179   digitalWrite(s1_trigg, LOW);
180   delayMicroseconds(2);
181   digitalWrite(s1_trigg, HIGH);
182   delayMicroseconds(10);
183   digitalWrite(s1_trigg, LOW);
```

```
184  duration = pulseIn(s1_echo, HIGH);
185  distance= duration*0.034/2;
186   break;
187  ////////////////////////////////////
188  case 1:
189  digitalWrite(s2_trigg, LOW);
190  delayMicroseconds(2);
191  digitalWrite(s2_trigg, HIGH);
192  delayMicroseconds(10);
193  digitalWrite(s2_trigg, LOW);
194  duration = pulseIn(s2_echo, HIGH);
195  distance= duration*0.034/2;
196   break;
197  ////////////////////////////////////
198  case 2:
199  digitalWrite(s3_trigg, LOW);
200  delayMicroseconds(2);
201  digitalWrite(s3_trigg, HIGH);
202  delayMicroseconds(10);
203  digitalWrite(s3_trigg, LOW);
204  duration = pulseIn(s3_echo, HIGH);
205  distance= duration*0.034/2;
206   break;
207   ////////////////////////////////////
208  case 3:
209  digitalWrite(s4_trigg, LOW);
210  delayMicroseconds(2);
211  digitalWrite(s4_trigg, HIGH);
212  delayMicroseconds(10);
213  digitalWrite(s4_trigg, LOW);
214  duration = pulseIn(s4_echo, HIGH);
215  distance= duration*0.034/2;
216  break;
217   case 4:
218  digitalWrite(s5_trigg, LOW);
219  delayMicroseconds(2);
220  digitalWrite(s5_trigg, HIGH);
221  delayMicroseconds(10);
222  digitalWrite(s5_trigg, LOW);
```

```
223   duration = pulseIn(s5_echo, HIGH);
224   distance= duration*0.034/2;
225   break;
226   }
227   return distance;
228 }
229 //funkcija za detekciju stanja infracrvenih senzora
230 int IRsensor(int a)
231 {
232   int stanje=0;
233   switch(a)
234   {
235     case 0:
236        if (digitalRead(IRout1) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
237     case 1:
238        if (digitalRead(IRout2) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
239     case 2:
240        if (digitalRead(IRout3) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
241     case 3:
242        if (digitalRead(IRout4) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
243     case 4:
244        if (digitalRead(IRout5) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
245     case 5:
246        if (digitalRead(IRout_left) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
247     case 6:
248        if (digitalRead(IRout_right) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
249     case 7:
250        if (digitalRead(IRout_back) == LOW)    {  stanje=1;  }  else  {
     stanje=0; } break;
251   }
252   return stanje;
253 }
```

```
254 void setup() {
255   pinMode(Vout, INPUT);
256   sv.attach(5);
257  ////////////////////////////////////
258   pinMode(s1_trigg,OUTPUT);
259   pinMode(s1_echo,INPUT);
260   pinMode(s2_trigg,OUTPUT);
261   pinMode(s2_echo,INPUT);
262   pinMode(s3_trigg,OUTPUT);
263   pinMode(s3_echo,INPUT);
264   pinMode(s4_trigg,OUTPUT);
265   pinMode(s4_echo,INPUT);
266   pinMode(s5_trigg,OUTPUT);
267   pinMode(s5_echo,INPUT);
268   pinMode(lamp , OUTPUT);
269   //////////postava za motor driver/////////
270   pinMode(MOTOR_A1_PIN, OUTPUT);
271   pinMode(MOTOR_B1_PIN, OUTPUT);
272   pinMode(PWM_MOTOR_1, OUTPUT);
273   pinMode(EN_PIN_1, OUTPUT);
274   ///////postava infracrvenih senzora//////////////
275   pinMode(IRout1,INPUT);
276   pinMode(IRout2,INPUT);
277   pinMode(IRout3,INPUT);
278   pinMode(IRout4,INPUT);
279   pinMode(IRout5,INPUT);
280   pinMode(IRout_left,INPUT);
281   pinMode(IRout_right,INPUT);
282   pinMode(IRout_back,INPUT);
283   ////////////////////////////////////
284   //dio koji se odnosi samo za izracunavanje i ocitavanje ziroskopa//
285   Serial.begin(9600);
286   Wire.begin(); // inicijalizacija Wire biblioteke
287   // postava ADXL345 u stanje mjerenja
288   Wire.beginTransmission(ADXL345); // pocetak komunikacije sa uredajem
289   Wire.write(0x2D); // komunikacija sa POWER_CTL Registrom - 0x2D
290   // omoguci mjerenje
291   Wire.write(8); // Bit D3 High za mjerenje je  omogucen (8dec -> 0000
       1000 binary)
```

```
292   Wire.endTransmission();
293   delay(10);
294   //Off-set Calibration
295   //X-axis
296   Wire.beginTransmission(ADXL345);
297   Wire.write(0x1E);
298   Wire.write(1);
299   Wire.endTransmission();
300   delay(10);
301   //Y-axis
302   Wire.beginTransmission(ADXL345);
303   Wire.write(0x1F);
304   Wire.write(-2);
305   Wire.endTransmission();
306   delay(10);
307   //Z-axis
308   Wire.beginTransmission(ADXL345);
309   Wire.write(0x20);
310   Wire.write(-9);
311   Wire.endTransmission();
312   delay(10);
313 }
314 void loop() {
315   ////////////////////////////////
316   float struja;
317   float prosjekSonda=0;
318   int br=0;
319   int smjer;
320 digitalWrite(EN_PIN_1, HIGH);
321 //dio koda za paljenje i gasenje svijetla preko foto-otpornika//
322   int c = analogRead(A1);
323 if ( c>600){
324   digitalWrite(37,HIGH);
325 }
326 else if ( c <600 ){
327 digitalWrite(37,LOW);
328 }
329 Serial.print("");
330 //
```

```
    ////////////////////////////////////////////////////////////////////////////////////////////////////

//sa ovom petljom realiziramo polje stanja za infracrvene senzore te ih isto ispisujemo//
int brojilo=0;

for(int j=0;j<=7;j++)
{
  IRsensorARRAY[j]=IRsensor(j);
}
for(i=0;i<=7;i++)
{
  Serial.print(IRsensorARRAY[i]);
}

///dio za inicijalizaciju polja udaljenosti//
   for(i=0;i<=4;i++)
   {
     distance_array[i]=distance_calc(i);
   }
//za inicijalizaciju giroskopa//
  Wire.beginTransmission(ADXL345);
  Wire.write(0x32); // zapo ni sa registrom 0x32 (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(ADXL345, 6, true); // pro itaj svih 6 registara, svaka os je pospremljena u 2 registra
  X_out = ( Wire.read() | Wire.read() << 8); //  itanje  x-osi
  X_out = X_out / 256; //vrijednosti se dijele sa 256 prema podacima
  Y_out = ( Wire.read() | Wire.read() << 8); //  itanje  y-osi
  Y_out = Y_out / 256;
  Z_out = ( Wire.read() | Wire.read() << 8); //  itanje  z-osi
  Z_out = Z_out / 256;
  // izra unavanje nagiba u x i y osi
  roll = atan(Y_out / sqrt(pow(X_out, 2) + pow(Z_out, 2))) * 180 / PI;
  pitch = atan(-1 * X_out / sqrt(pow(Y_out, 2) + pow(Z_out, 2))) * 180 / PI;

/// izracun akceleracije i udaljenosti//

```

```
365 //za ovo budemo isto tako koristili i  iroskop  koji ce odredivati
      brzinu ovisno nagibu.
366 //.jer ce trebat vise snage da se savladaju nagibi
367 speed_control(Y_out);
368 smjer=odabir_smjera(distance_array);
369
370  //nacin kontrole samog motora i sustav izbjegavanja
371  if(distance_array[2]>20&&distance_array[1]>10&&distance_array[3]>10&&
      IRsensorARRAY[0]==0&&IRsensorARRAY[1]==0&&IRsensorARRAY[2]==0&&
      IRsensorARRAY[3]==0&&IRsensorARRAY[4]==0)
372   {
373     forward();
374     ////////////////////////////////////////////////////
375         double voltagem=0;                             //
376       double ampsm=0;                                  //
377                                                        //
378             RawValue = analogRead(analogIn);           //
379             Voltage = ((RawValue / 1024.0) * 5000);    /////ako se
      koristi senzor struje
380                                                        //
381             Amps = (( Voltage - ACSoffset) / mVperAmp);//
382             Serial.print("Struja je:");                //
383             Serial.print(Amps);                        //
384     ////////////////////////////////////////////////////
385    // delay(700);
386     //brake();
387   }
388    else
389   {
390     //ova logika se odnosi na sustav izbjegavanja kada se objekti nalaze
      pored vozila
391     if(smjer==1||IRsensorARRAY[5]==1){
392     right_30();
393     millis();
394     manevar_okretanja_45();
395     left_30();
396     millis();
397     brake();
398     }
```

```
399     else if(smjer==2){
400     right_15();
401     millis();
402     manevar_okretanja_45();
403     left_15();
404     millis();
405     brake();
406 
407     }
408    else if(smjer==3||(IRsensorARRAY[5]==1&&IRsensorARRAY[6]==1)){
409     straight();
410     millis();
411     manevar_okretanja_45();
412     millis();
413     brake();
414     }
415    else if(smjer==4||IRsensorARRAY[6]==1){
416     left_15();
417     millis();
418     manevar_okretanja_45();
419     right_15();
420     millis();
421     brake();
422     }
423    else if(smjer==5||IRsensorARRAY[6]==1){
424     left_30();
425     millis();
426     manevar_okretanja_45();
427     right_30();
428     millis();
429     brake();
430     }
431  }
432     for(i=0;i<=4;i++)
433  {
434  Serial.println("");
435  Serial.print("Distance");
436  Serial.println(i);
437  Serial.println(distance_array[i]);
```

```
438   Serial.println("");
439   }
440 ////////dio za brojanje vremena od pocetka programa i regisrira vrijeme
      na kraju svake iteracije////
441 Serial.print("Time:");
442 time=millis;
443 time_float+=(time);////700 je dodano radi zastajkivanja motora radi
      ocuvanja komponenti, a u slucaju da se radi o kontinuiranom gibanju
      700 se brise..==0.7 sec
444 Serial.print(time_float);
445 Serial.println();
446
447 }
```