Automatizirani čistač podova sa sustavom detekcije zone kretanja temeljem analize boje

Murić, Filip

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:793570

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-21**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij Automatika

AUTOMATIZIRANI ČISTAČ PODOVA SA SUSTAVOM DETEKCIJE ZONE KRETANJA TEMELJEM ANALIZE BOJE

Završni rad

Filip Murić

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1S: Obrazac za ocjenu završnog rada na stručnom prijediplomskom studiju

Ocjena završnog rada na stručnom prijediplomskom studiju

Ime i prezime pristupnika:	Filip Murić
Studij, smjer:	Stručni prijediplomski studij Elektrotehnika, smjer Automatika
Mat. br. pristupnika, god.	A 4561, 19.07.2019.
JMBAG:	0165080820
Mentor:	prof. dr. sc. Tomislav Keser
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	doc. dr. sc. Tomislav Galba
Član Povjerenstva 1:	prof. dr. sc. Tomislav Keser
Član Povjerenstva 2:	izv. prof. dr. sc. Alfonzo Baumgartner
Naslov završnog rada:	Automatizirani čistač podova sa sustavom detekcije zone kretanja temeljem analize boje
Znanstvena grana završnog rada:	**Procesno računarstvo (zn. polje računarstvo)**
Zadatak završnog rada:	Projektirati, izraditi i testirati sustav za automatizirano čišćenje podova u obliku mobilne platforme s rotirajućim četkama. Sustav treba imati mogućnost samostalnog kretanja prostorom i izbjegavati prepreke. Isto tako, sustav se mora moći kretati samo unutar zadanog područja/zone čišćenja koja se sadaje bojom definiranom i nacrtanom preprečnom strukturom - crtom. Isto tako omogućiti kretanje površinom prema zadanoj boji ili bez zadavanje iste. Upravljanje radom sustava i podešenje parametara omogućiti putem mobilne aplikacije ili web sučelja.

Datum ocjene pismenog dijela završnog rada od strane mentora:	17.09.2024.
Ocjena pismenog dijela završnog rada od strane mentora:	Izvrstan (5)
Datum obrane završnog rada:	30.09.2024.
Ocjena usmenog dijela završnog rada (obrane):	Izvrstan (5)
Ukupna ocjena završnog rada:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio stručni prijediplomski studij:	30.09.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O IZVORNOSTI RADA

Osijek, 30.09.2024.

Ime i prezime Pristupnika:	Filip Murić
Studij:	Stručni prijediplomski studij Elektrotehnika, smjer Automatika
Mat. br. Pristupnika, godina upisa:	A 4561, 19.07.2019.
Turnitin podudaranje [%]:	9

Ovom izjavom izjavljujem da je rad pod nazivom**: Automatizirani čistač podova sa sustavom detekcije zone kretanja temeljem analize boje**

izrađen pod vodstvom mentora prof. dr. sc. Tomislav Keser

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

SADRŽAJ

1. UVOD 1
1.1 Zadatak i struktura rada 2
2. SUSTAV ZA AUTOMATSKO ČIŠĆENJE PODOVA 3
2.1. Teorijski osvrt na problem i rješenje u sustavima za automatsko čišćenje podova 3
2.2. Prijedlog sklopovskog rješenja 5
2.3. Prijedlog programskog rješenja 8
3. REALIZACIJA AUTOMATIZIRANOG ČISTAČA PODOVA 10
3.1. Korištene komponente i programska okruženja 10
3.2. Realizacija sklopovskog rješenja 11
3.3. Realizacija programskog rješenja 18
4. TESTIRANJE I REZULTATI 23
4.1. Metodologija testiranja 23
4.2. Rezultati testiranja 23
5. ZAKLJUČAK 30
LITERATURA 32
SAŽETAK 33
ABSTRACT 34
ŽIVOTOPIS 35
PRILOZI I DODACI 36

1. UVOD

Od početka ljudske civilizacije ljudi teže tome da si što više olakšaju život uporabom raznih pomagala i alata. Kako tehnologija napreduje tako je čovjekov život i poslovi koje obavlja sve lakši i lakši. Takva se praksa proširila svim ljudskim djelatnostima, prvo u tvornicama gdje su tako stvorene prve pokretne trake, robotske ruke, kako bi proces proizvodnje bio efikasniji i lakši. Tako u današnjem svijetu postoji sve veća potreba za unaprjeđenjem tehnologije i računalnih sustava. Jedan od takvih pothvata kojima je cilj bio olakšati život ljudi jest i izum automatiziranog mobilnog sustava za čišćenje podova.

Automatizirani čistači podova postaju jedan od najpopularnijih kućanskih aparata, uvelike olakšavajući naše živote jer automatizira jedan aspekt kućanskih poslova što nam omogućuje da produktivnije provedemo to vrijeme koje bi proveli čisteći [1]. Većina modela automatiziranih čistača podova ima mogućnost povezivanja s mobilnim uređajima preko kojih se s njim može upravljati a može se i nadzirati njihov rad. Također se može odrediti vrijeme čišćenja kao i zone koje je potrebno očistiti ili koje nije potrebno čistiti unutar prostorije. Zahvaljujući tome njih se može beskontaktno pokretati, pratiti te upravljati na daljinu. Čišćenje se provodi s vakumskom pumpom sličnoj onoj u konvencionalnim usisavačima, vakuum pumpa manje je veličine od promjera samog čistača stoga se koriste i rotirajuće metlice koje guraju prljavštinu u vakumsku pumpu, zahvaljujući metlicama čistač lako čisti kutove. Određeni modeli imaju nastavak koji koristi vodu kako bi otklonio mrlje s poda.

Automatizirani čistači podova koriste se raznim senzorima kako bi detektirali i izbjegli prepreke i što efikasnije i brže očistili prostoriju. U branik čistača ugrađeni su optički prekidači koji detektiraju koliziju s preprekama. Najefikasniji i najbrži put kojim bi se trebao kretati su paralelne linije, kako bi to postigao čistač posjeduje žiroskope i senzore na kotačima kako bi se precizno okrenuo pod željenim kutom. Koristi *vSLAM* tehnologiju pomoću koje stvara dvodimenzionalnu mapu prostorije u kojoj čisti, razlikuje zidove od manjih prepreka kao što su stolice i ažurira put oko detektiranog objekta kako bi i dalje išao najefikasnijim putem. *vSLAM* je proces mapiranja prostorije koji se izvršava na način da se izračunava pozicija i orijentacija kamere u odnosu na okolinu, proces koristi samo vizualne inpute kamere [2]. Posjeduje infracrvene detektore i svijetleće diode uperene prema podu pomoću kojih detektira razliku u visini kako bi se pravovremeno zaustavio kada naiđe na stepenice. Skuplji

modeli imaju 3d senzor dubine koji služi za detekciju kompleksnijih prepreka kao što su kablovi ili odjeća, 3d senzor je puno precizniji u detekciji prepreka od *vSLAM-a* ali nema mogućnost mapiranja prostorije. Čistač ima mogućnost vraćanja u bazu na punjenje kada detektira da će se baterija isprazniti. Noviji modeli imaju mogućnost automatskog pražnjenja vrećice za prašinu u vrećice u bazi koje je potrebno mijenjati jednom mjesečno što uvelike smanjuje količinu vremena potrebnog za održavanje čistača.

1.1 Zadatak i struktura rada

Zadatak ovog rada je stvoriti funkcionalan prototip automatskog čistača podova koji će biti sposoban prepoznavati i zaobilaziti različite prepreke, samostalno se kretati prostorom, efikasno uklanjati prljavštinu i prašinu, te samostalno manevrirati prostorom uz pomoć raznih senzora. Potrebno je omogućiti upravljanje radom automatskog čistača putem mobilne aplikacije.

U drugom poglavlju objašnjeni su teoretski prijedlozi rješenja sklopovskog i programskog djela završnoga rada dok se u trećem poglavlju dana konkretna rješenja sklopovskog i programskog djela završnog rada, detaljno su opisane korištene komponente i programski alati korišteni za izradu čistača. U četvrtom poglavlju obavljena su testiranja, opisane su metode testiranja koje su korištene za provjeru funkcionalnosti i performansi automatskog čistača podova i rezultati testiranja dok se u petom poglavlju nalazi zaključak u kojem se sažimaju postignuti rezultati i ispunjenje ciljeva.

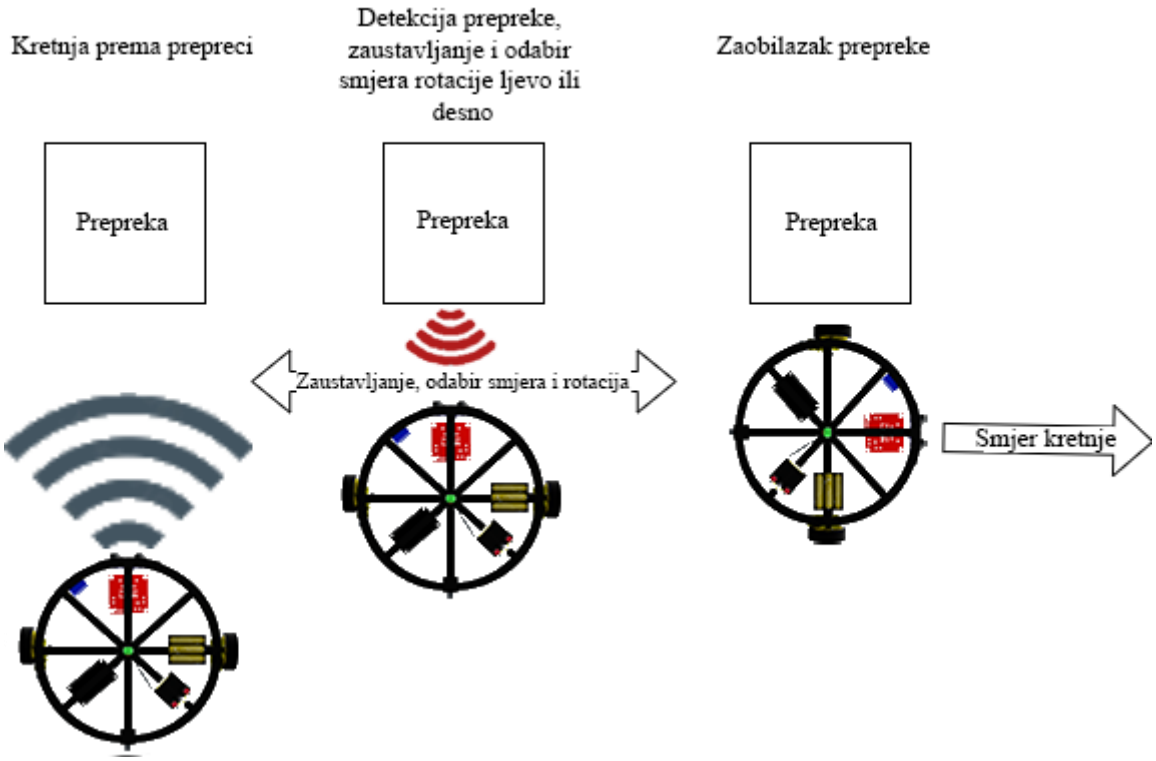
2. SUSTAV ZA AUTOMATSKO ČIŠĆENJE PODOVA

2.1. Teorijski osvrt na problem i rješenje u sustavima za automatsko čišćenje podova

Problemi su podijeljeni na 3 djela. To su detekcija prepreka na putu i način na koji čistač na njih reagira, uvjet da čistačem upravljamo putem mobilne aplikacije te definiranje prostora unutar kojega čistač čisti crtom u boji.

2.1.1. Prepoznavanje i izbjegavanje prepreka

Prvi problem koji se pojavljuje je detekcija prepreka i izbjegavanje istih kako ne bi došlo do sudara između čistača i prepreke. Ultrazvučni senzor koristiti će se za mjerenje udaljenosti kako bi čistač pravovremeno mogao stati i ne sudariti se s preprekom te uz pomoć elektromotora koji imaju mogućnost vrtnje u oba smjera čistač će se okretati i ići u drugom smjeru, te će se nastavljati kretati sve dok ne naiđe na sljedeću prepreku. Ultrazvučni senzori funkcioniraju na način da šalju ultrazvučne valove frekvencija iznad 20kHz što je gornja granica raspona frekvencija koje ljudsko uho može detektirati. Ti se valovi šalju ispred senzora te ako pogode prepreku valovi se odbijaju nazad prema senzoru. Na osnovu proteklog vremena od kada se ultrazvučni val poslao do kada je detektiran odbijeni ultrazvučni val vrši se izračun udaljenosti detektirane prepreke. Pri radu s ultrazvučnim senzorima potrebno je voditi računa o tome kroz koji medij ultrazvučni valovi prolaze.



Slika 2.1. Prikaz rješenja problema izbjegavanja prepreka.

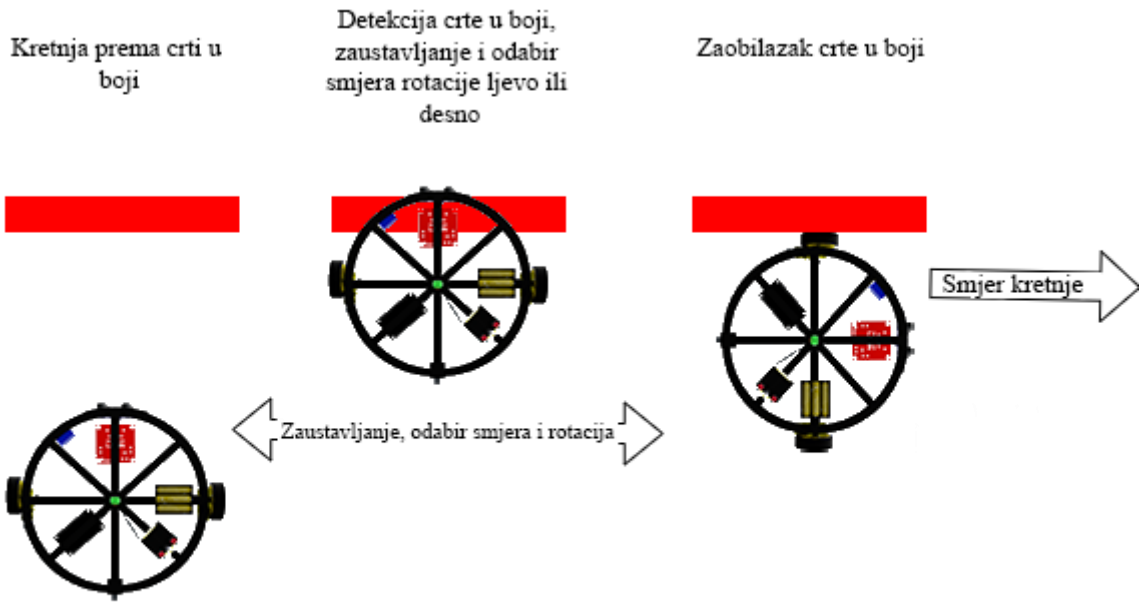
2.1.2. Upravljanje čistačem pomoću mobilne aplikacije

Drugi problem je uvjet da čistačem upravljamo putem mobilne aplikacije. Izabrani način povezivanja je *bluetooth*, potrebno je odabrati odgovarajući mikro kontroler koji ima mogućnost povezivanja *bluetooth*-om. Izrada mobilne aplikacije predstavlja problem zbog naizgled visokog stupnja složenosti. Izabrana je online besplatna platforma *MIT App Inventor* zbog svoje jednostavnosti i mogućnosti blokovskog kodiranja [3].

2.1.3. Kretanje unutar granica definiranih bojom

Ukoliko korisnik želi čistaču ograničiti područje čišćenja na nekakav manji dio u prostoriji to može postići tako da na pod postavi crte u boji. Zahvaljujući senzoru boja čistač će moći detektirati boju na podu te svoju kretnju ograničiti samo na to područje koje je omeđeno bojom. U tom režimu rada čistač obojane crte tretira kao i prepreke. Ovaj režim rada je koristan kada korisnik ima samo manji dio prostorije koji treba očistiti. Jedan od problema koji automatski čistač podova mora savladati

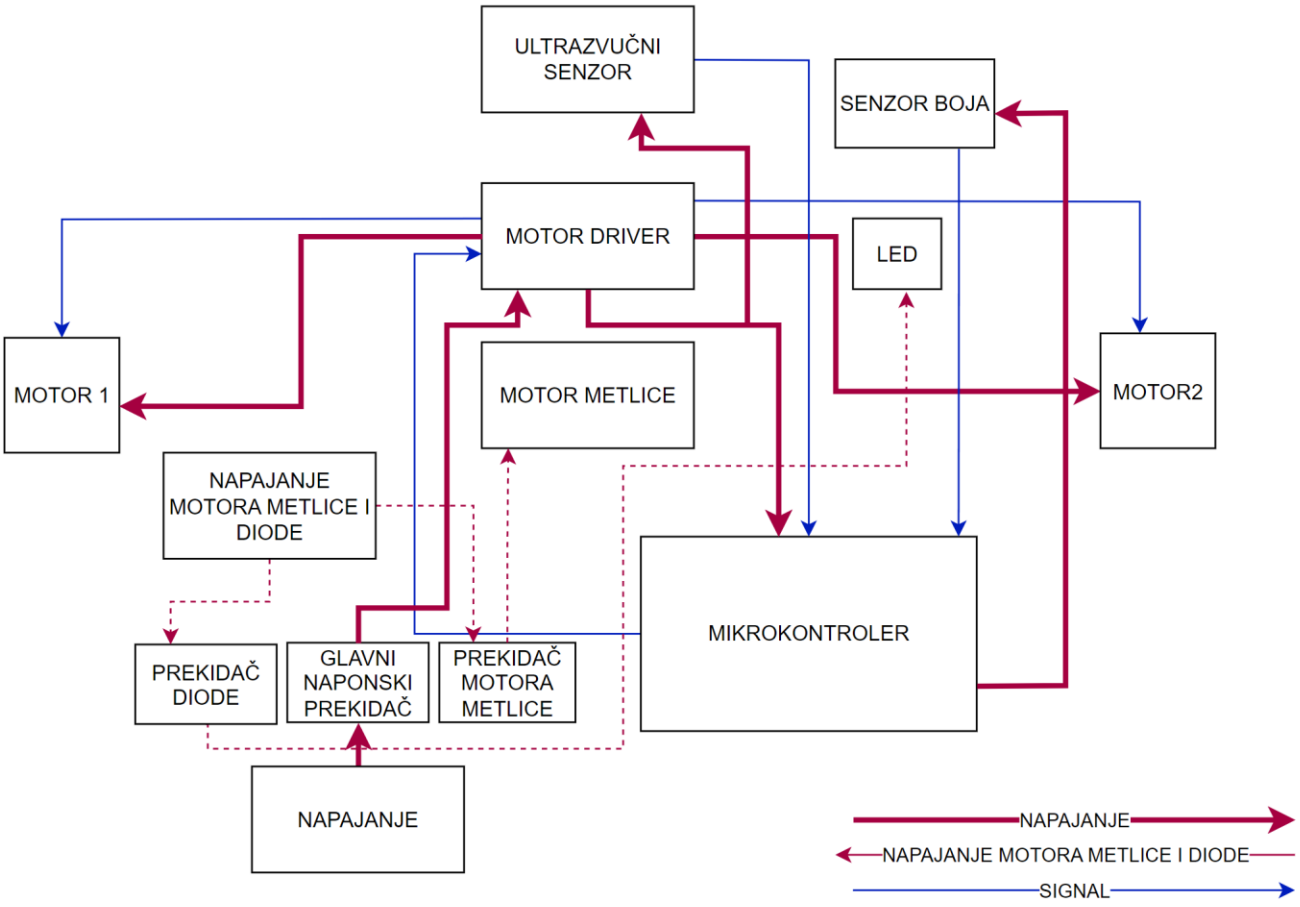
je detekcija stepenica kako ne bi pao niz njih. To isto tako može biti riješeno postavljanjem crte u boji na rub stepenica. Senzori boja posjeduju fotodetektor koji ima mogućnost prepoznavanja osnovne 3 boje: crvena, zelena i plava. Kada svijetlost obasja objekt on reflektira svoju boju a sve druge boje upije te je tu boju onda moguće detektirati fotodetektorom. Ukoliko se radi o kompleksnijoj boji odnosno mješavini osnovnih boja onda senzor izračunava udio svake boje.



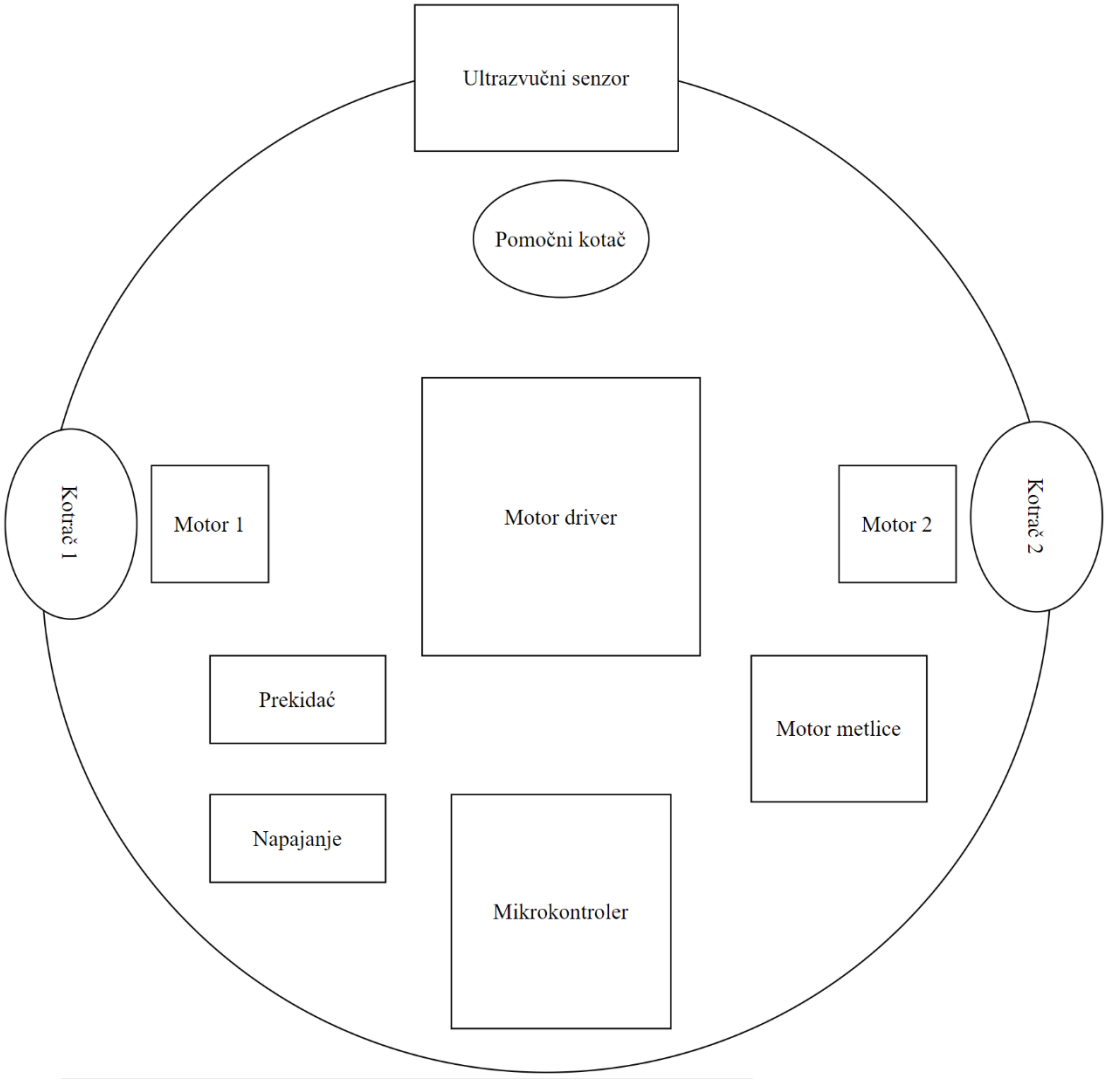
Slika 2.2. Prikaz detekcije crte u boji i reakcija na crtu u boji .

2.2. Prijedlog sklopovskog rješenja

Mikroupravljač će pomoću vrijednosti dobivenih od ultrazvučnog senzora i senzora boja upravljati kretanjem čistača preko motor *driver*-a. Motor *driver* će napajati mikroupravljač, ultrazvučni senzor te motore. Senzor boja napajati će se preko mikroupravljača. Između napajanja i motor *driver*-a nalazi se prekidač. Motor metlice je stalno uključen. Čistač će za kretanje koristiti dva pogonska kotača spojena s osovinom istosmjernih motora i jedan pomoćni kotač. Osmišljeni način povezivanja komponenti vidljiv je na slici 2.3.



Slika 2.3. Strukturna shema.

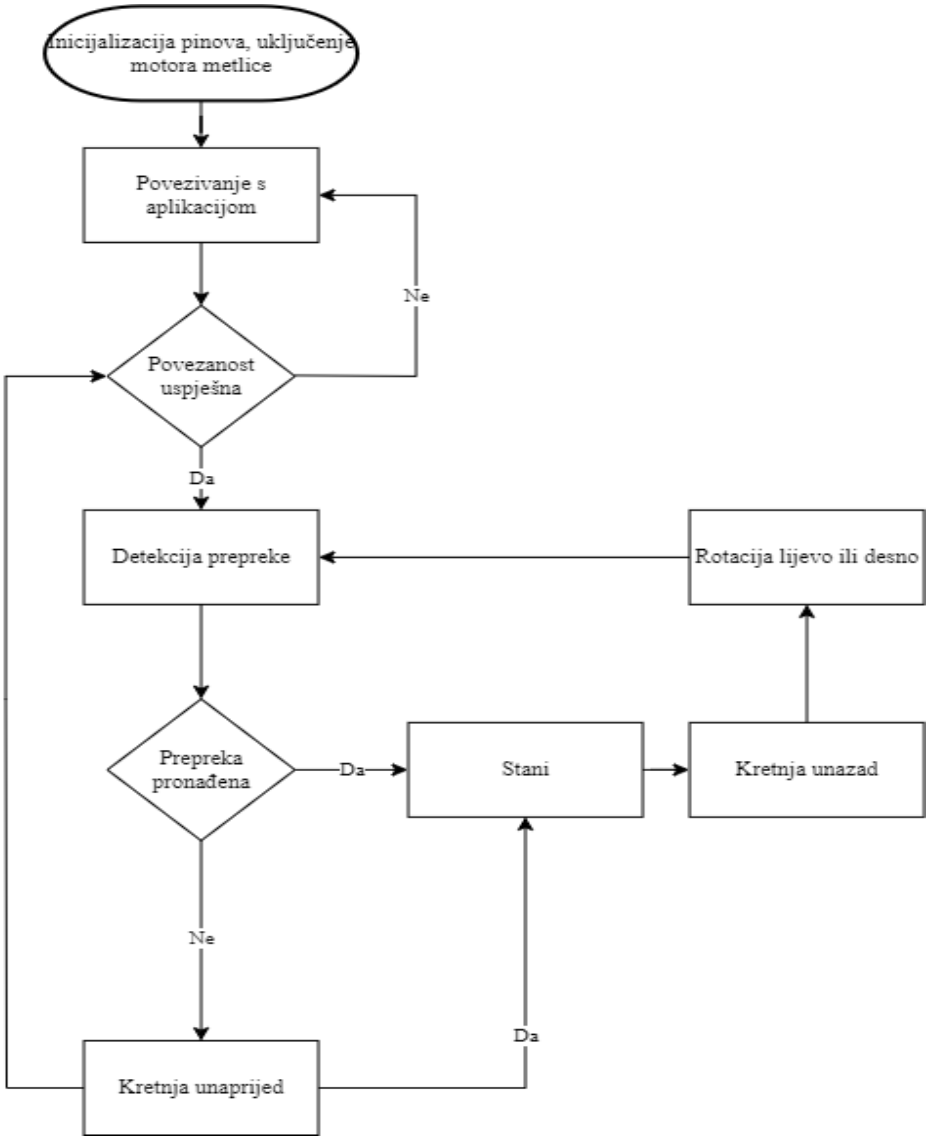


Slika 2.4. Prikaz raspodijele komponenata unutar kućišta čistača.

Iz slike 2.4. možemo vidjeti zamišljenu raspodjelu komponenata unutar kućišta čitača. Položaj ultrazvučnog senzora nalazio bi se na samom početku čistača kako bi pravovremeno detektirao prepreke a mikroupravljač na samom kraju kako bi njegov utor za *micro USB (eng. universal serial bus, hrv. univerzalna serijska sabirnica, u daljnjem tekstu korišten kao USB)* kabel ostao pristupačan za laku konekciju s računalom u slučaju potrebe dorade koda. Izvor napajanja je punjiva baterija koju je lako izvaditi iz kućišta. Naponski prekidač nalaziti će se na gornjem dijelu kućišta za lako paljenje i gašenje čistača. Pomoćni kotač nalaziti će se na prednjem dijelu kućišta kako čistač ne bi ribao po podu. Motor *driver* biti će smješten u sredini kućišta iz razloga što se s njim spaja najviše vodiča. Kućište čistača će biti izrađeno od 3D isprintane plastike što omogućava veliku fleksibilnost s odabirom dimenzije kućišta i raspodjelom komponenti.

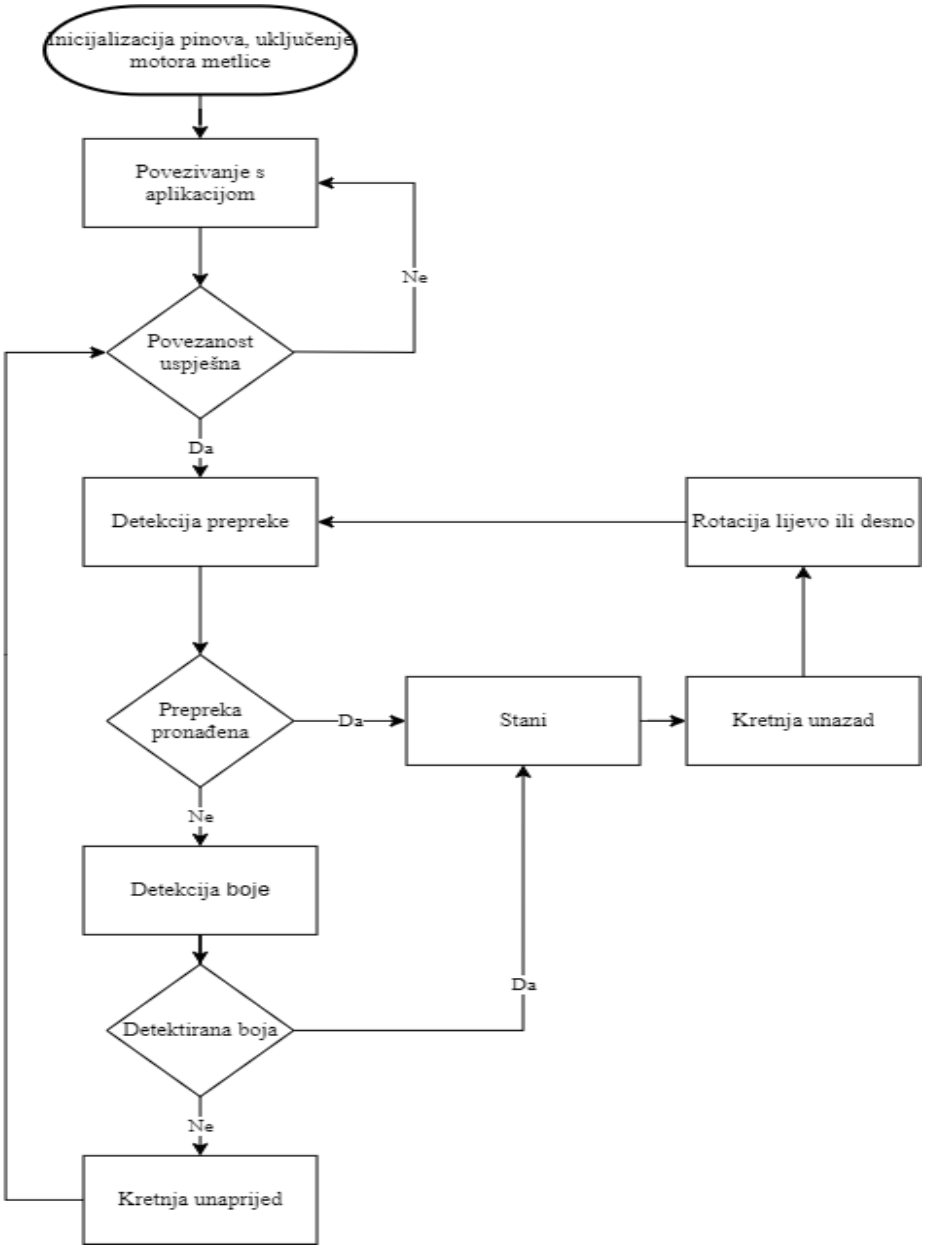
2.3. Prijedlog programskog rješenja

Na početku programa inicijaliziraju se izlazni i ulazni pinovi i uključuje motor metlice. Nakon uspješnog povezivanja s aplikacijom iščitava se udaljenost od prepreke, ako je prepreka detektirana oba motora će se zaustaviti, zatim će se zavrtiti u suprotnom smjeru kako bi se odmaknuli od prepreke te će se rotirati lijevo ili desno te se naposljetku ponovno izvršava detekcija prepreke. Ako prepreka nije pronađena motori se vrte unaprijed. Iz slike 2.5. možemo vidjeti kako se u petlji stalno vrši provjera povezanosti s aplikacijom te detekcija prepreka.



Slika 2.5. Dijagram toka programa bez senzora boja.

U režimu rada u kojem je automatizirani čistač prostorno ograničen s bojom princip rada je isti kao i u režimu rada bez ograničenosti bojom samo što se vrši dodatna provjera. Ukoliko je detektirana boja motori se zaustavljaju, započinju se vrtjeti u suprotnom smjeru te se rotiraju lijevo ili desno te se ponovno vrši provjera udaljenosti i detekcija boje kao što možemo vidjeti na slici 2.6.. U principu automatizirani čistač detektiranu boju vidi kao prepreku.



Slika 2.6. Dijagram toka programa sa senzora boja.

3. REALIZACIJA AUTOMATIZIRANOG ČISTAČA PODOVA

3.1. Korištene komponente i programska okruženja

Za izradu čistača korištene su sljedeće komponente:

mikroupravljač ESP32 Wroom [P.4.]

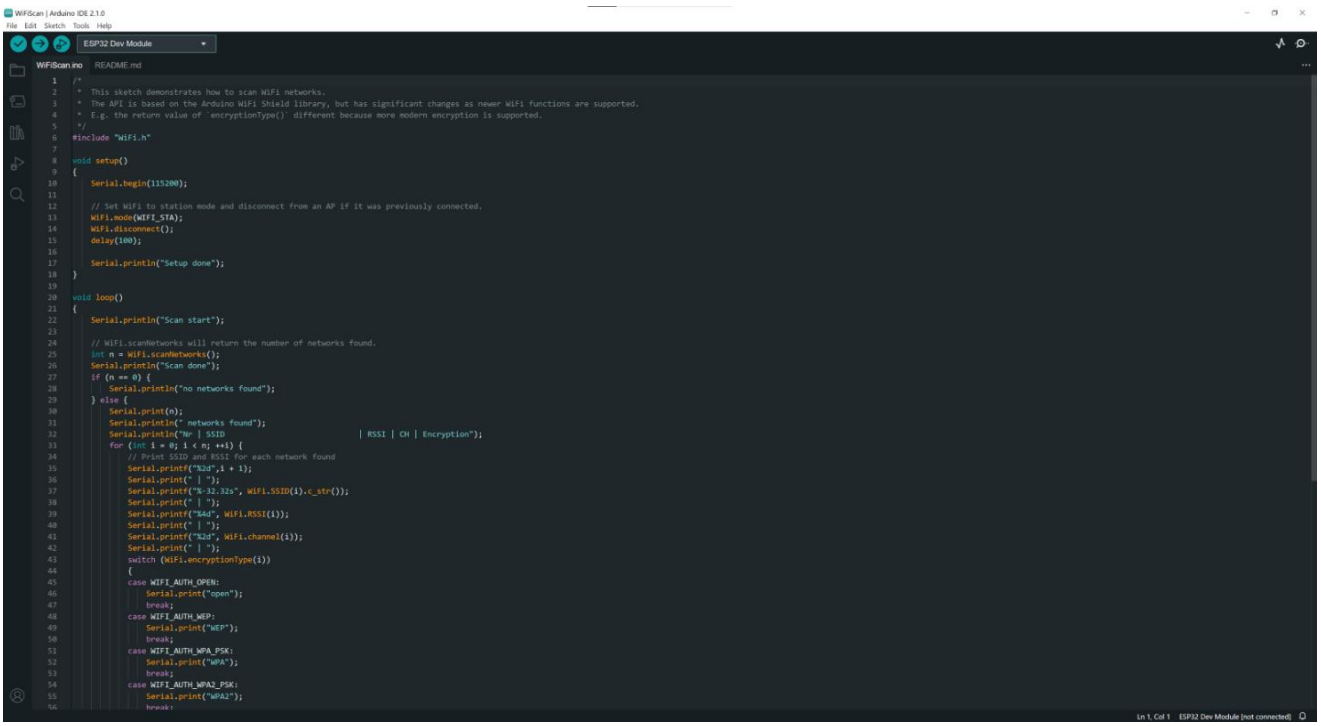
motor *driver* L298N [P.2.]

ultrazvučni senzor HC-SR04 [P.1.]

punjiva baterija od 9 volti te 3 serijski spojene baterije od 1.5 volti koje služe za pokretanje motora četkice [P.5.]

senzor za prepoznavanje boja APDS9960 [P.3.]

dva istosmjerna motora za kretanje te jedan istosmjerni motor koji služi za rotaciju četkice čistača. [P.6.]

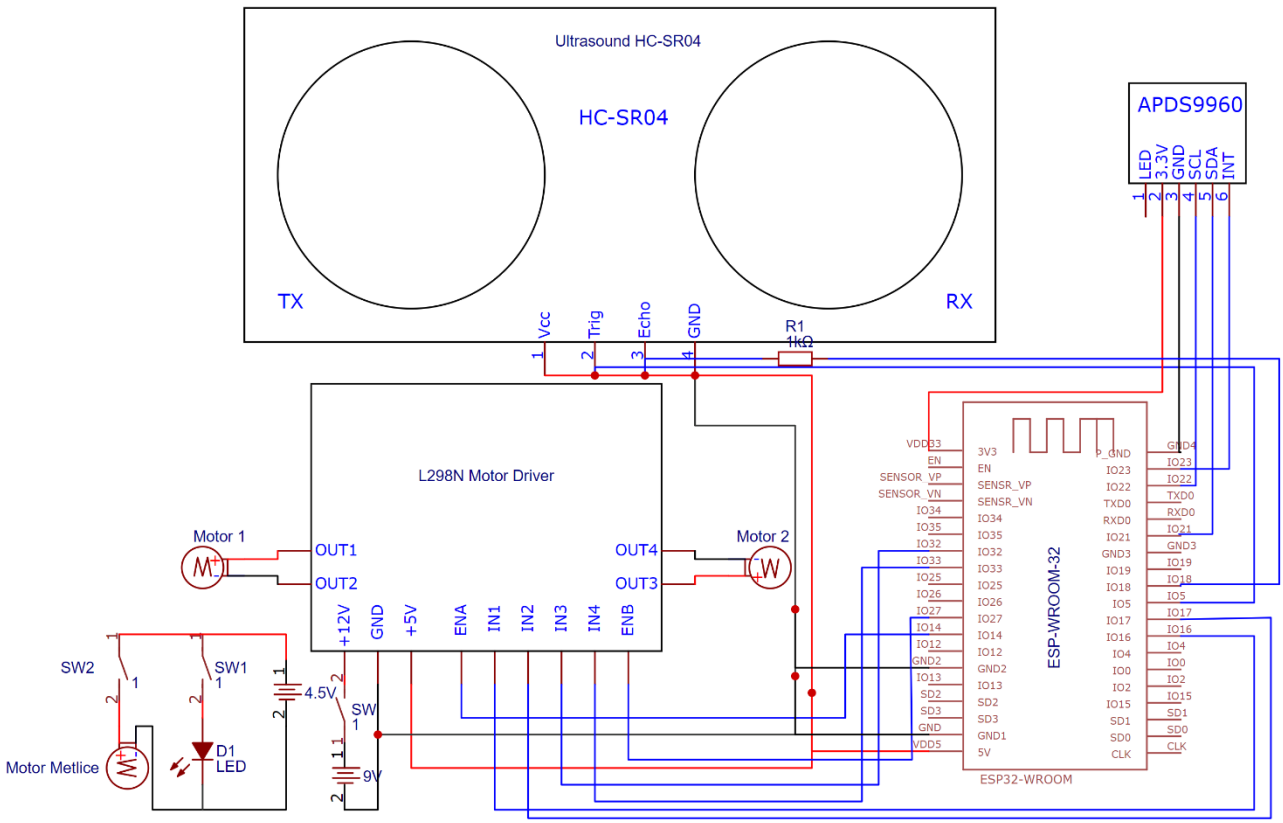


Slika 3.1. Arduino IDE sučelje.

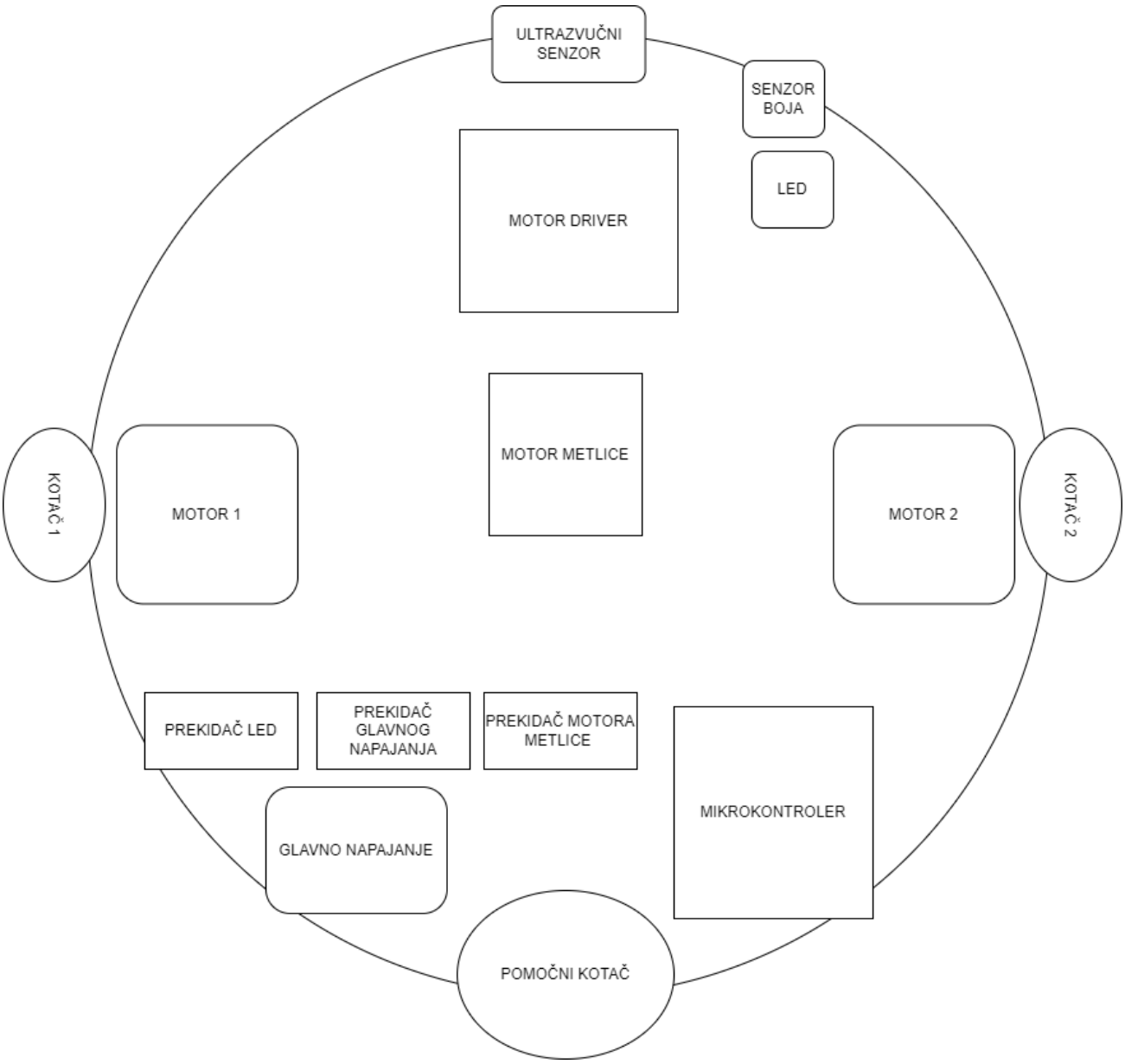
Za programiranje ESP32 korištena je razvojna aplikacija *Arduino IDE(eng integrated development environment, hrv. Integrirano razvojno okruženje, u daljnjem tekstu korišten kao IDE)* u kojoj se koriste programski jezici C i C++. *Arduino IDE* izabran je zbog svoje jednostavnosti i popularnosti. Kako bi koristili određene senzore ili aktuatore dovoljno je samo preuzeti odgovarajući *library* za taj senzor ili aktuator. Postoji i mogućnost korištenja ugrađenog serijskog monitora i crtača za otklanjanje pogrešaka i vizualizaciju podataka. Za izradu mobilne aplikacije korišten je *MIT App Inventor*. To je online alat za izradu mobilnih aplikacija, vrlo je jednostavan i intuitivan za korištenje. Programira se blokovski, nakon izrade aplikacije na računalu potrebno je preuzeti aplikaciju *MIT AI2 Companion*, zatim se pomoću QR koda izrađena aplikacija lako prenese na mobilni uređaj [4].

3.2. Realizacija sklopovskog rješenja

Za napajanje se koristi punjiva baterija od 9 volti kapaciteta 200 mAh. Na slici 3.2. uočavamo da je baterija spojena na motor *driver* L298N preko prekidača. Motor *driver* ima izlaz od 5 volti preko kojega se napaja mikroupravljač ESP32 WROOM te ultrazvučni senzor HC-SR04. Ultrazvučni senzor posjeduje dva pina *Trig* i *Echo* pomoću kojih prenosi podatke o udaljenosti mikroupravljaču. Motor *driver* napaja i oba pogonska kotača sa približno 7 volti, do pada napona od 2 volta dolazi zbog naponskog regulatora unutar samoga motor *driver*-a. Korištena su 2 istosmjerna univerzalna motora koja rade na rasponu napona od 6 do 12 volti. Motor *driver* ima mogućnost regulacije napona pomoću *PWM*-a (eng. pulse width modulation, hrv. pulsno širinska modulacija, u daljnjem tekstu korišten kao *PWM*) , kako bi to postigao motor *driver* posjeduje pinove *ENA* i *ENB* koji se spajaju na mikroupravljač. Motor *driver* ima mogućnost upravljanja smjerom vrtnje oba motora pomoću pinova *IN1, IN2, IN3, IN4* koji su spojeni na mikroupravljač. Senzor boja apds9960 se napaja preko mikroupravljača i to naponom od 3.3 volta a komunicira s mikroupravljačem preko pinova *SCL* i *SDA* i za to koristi I^2C komunikaciju . Koristi se i bijela svijetleća dioda koja se nalazi odmah pokraj samoga senzora boja kako bi se osvijetlila površina podloge u mračnijim uvjetima rada, ona se napaja isto kao i motor metlice s 3 AAA baterije od 1.5 volta spojenih serijski. Motor metlice i svijetleća dioda imaju svaki svoj prekidač.



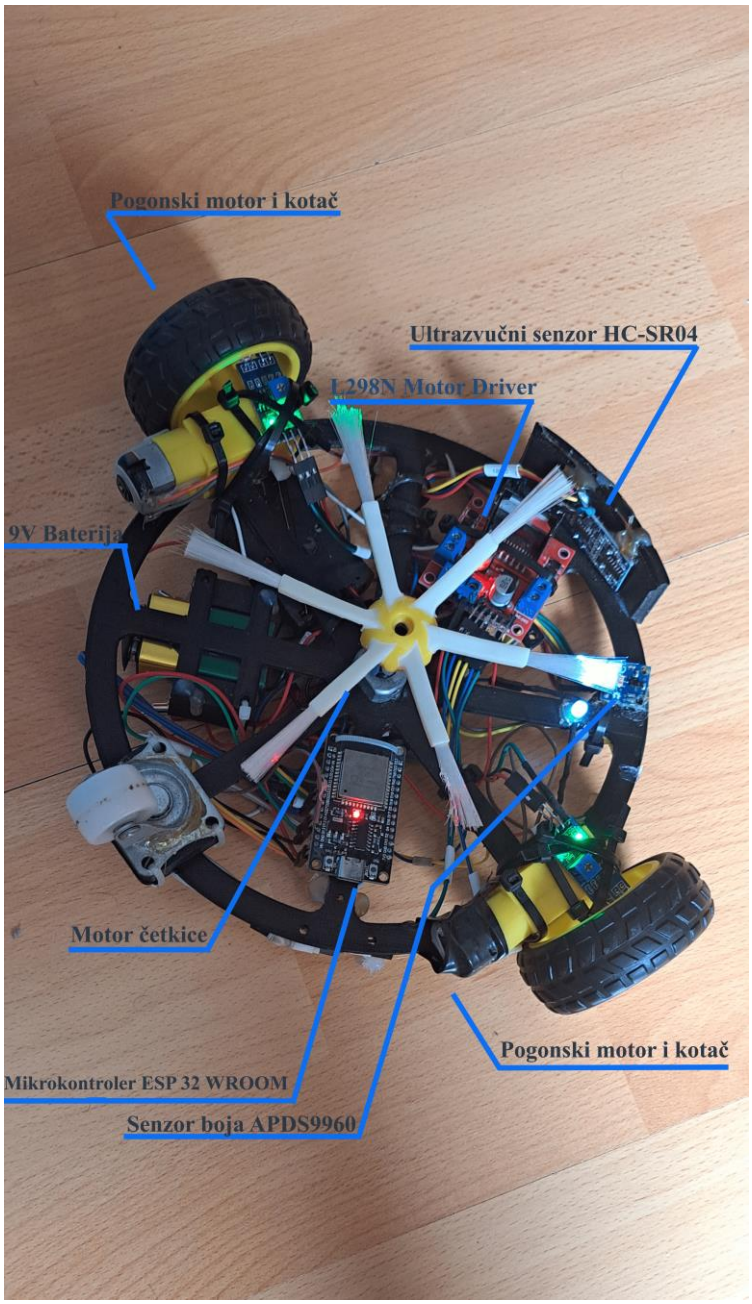
Slika 3.2. Električna shema čistača.



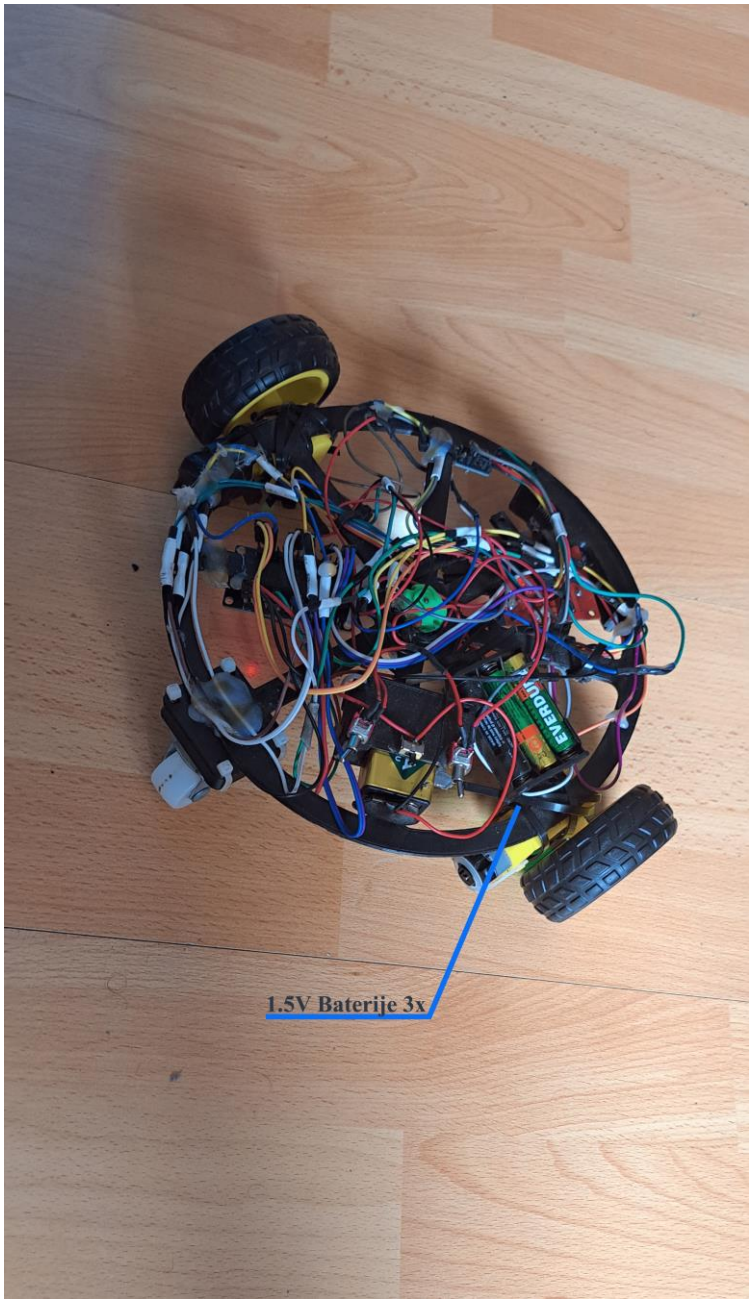
Slika 3.3. Pojednostavljeni prikaz raspodijele komponenata čistača.

Iz pojednostavljenog prikaza slike 3.3. uočavamo da je došlo do određenih promjena u odnosu na tehnički nacrt prikazan u poglavlju 2.2. Došlo je do preraspodijele određenih komponenata te su dodana dva prekidača jedan za svjetleću diodu a drugi za motor metlice. Dodan je i drugi izvor napajanja za svjetleću diodu i motor metlice. Razlog dodavanja drugog izvora napajanje je što svjetleća dioda i motor metlice zahtijevaju niže naponske razine od postojećih unutar glavnog

sklopovlja te kako bi se smanjilo opterećenje glavne baterije odnosno kako bi čistač imao što duže vrijeme uporabe između punjenja.

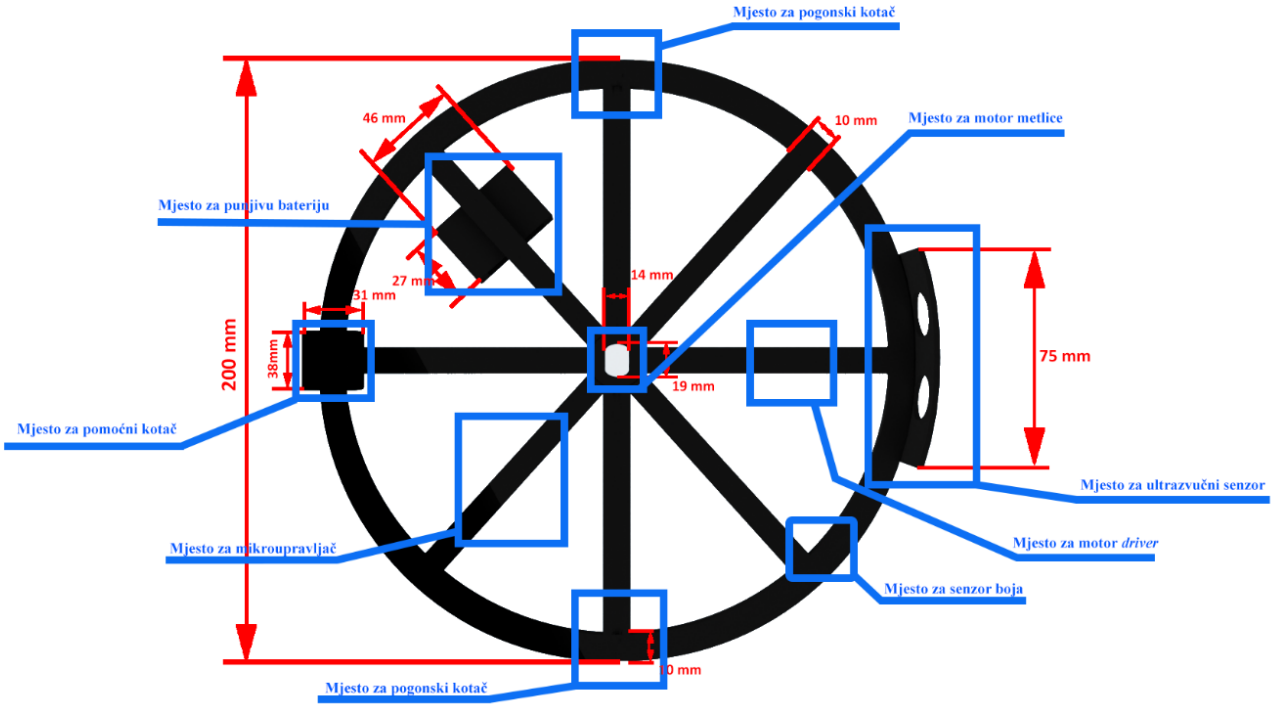


Slika 3.4. Prikaz donje strane čistača s označenim komponentama.

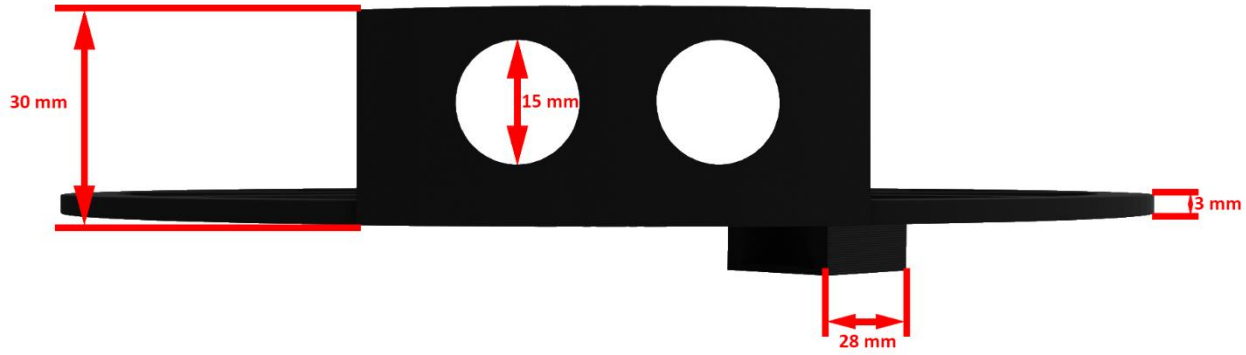


Slika 3.5. Prikaz gornje strane čistača s označenim komponentama.

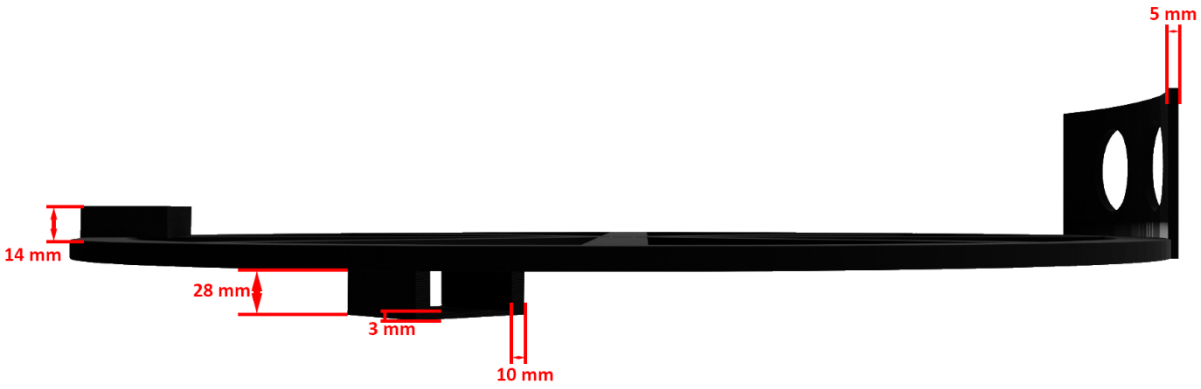
Na slici 3.4. i slici 3.5. jasno je prikazan finalni raspored komponenti do kojega je došlo zbog manjka prostora na kosturu čistača i balansiranja težine. Na kućištu punjive baterije nalaze se sva tri prekidača kako bi bili što dostupniji. Punjiva baterija te baterije motora metlice i svijetleće diode su vrlo pristupačne kako bi se s lakoćom mogle po potrebi zamijeniti odnosno napuniti u slučaju punjive baterije.



Slika 3.6. Tehnički nacrt kostura čistača s gornje strane.



Slika 3.7. Tehnički nacrt kostura čistača s prednje strane.



Slika 3.8. Tehnički nacrt kostura čistača s bočne strane.

Iz Slike 3.6., Slike 3.7. i Slike 3.8. uočavamo da je sama baza čistača izrađena od 4 komada 3D printane plastike. Najveći dio je kostur čistača promjera 20 centimetara i debljine 3 milimetra. S prednje strane kostura nalazi se zaštitnik ultrazvučnog senzora s otvorima. Sa stražnje strane kostura nalazi se baza za treći kotač koja je potrebna da bi se treći kotač iz nivelirao s 2 pogonska kotača. Posljednja 3D printana komponenta je držač punjive baterije od 9V koji je dizajniran tako da se baterija s lakoćom može izvaditi kada ju je potrebno napuniti. Sve tri 3D printane komponente su pričvršćene za kostur čistaća sa specijalnim ljepilom za plastiku.

3.3. Realizacija programskog rješenja

U ovom dijelu objašnjen je programski kod napisan u Arduino IDE-u i aplikacija rađena u *MIT App Inventor*-u. Na početku izvršavanja programskog koda pozivaju se biblioteke: *Wire* koja je potrebna za I2C komunikaciju koju koristi senzor boja, biblioteka *BluetoothSerial* koja se koristi za povezivanje i komunikaciju s aplikacijom , te biblioteka SparkFun_APDS9960 potrebna za rad senzora boja te se inicijaliziraju pinovi: dva pina potrebna za ultrazvučni senzor, dva pina potrebna za kontrolu brzine vrtnje motora pomoću *PWM*-a te četiri pina potrebna za upravljanje smjera vrtnje pogonskih motora. Na početku koda definirane su mnoge varijable i konstante. Zatim se u *setup* dijelu koda inicijalizira senzor boja, postavlja se brzina serijske komunikacije te se pinovi pogonskih motora postavljaju kao izlazi dok se kod ultrazvučnog senzora *triger* pin postavlja kao izlaz a *echo* kao ulaz. U dogovoru s mentorom odabrana je crvena boja kao boja kojom se može čistaću ograničiti zona čišćenja odnosno kretanja.

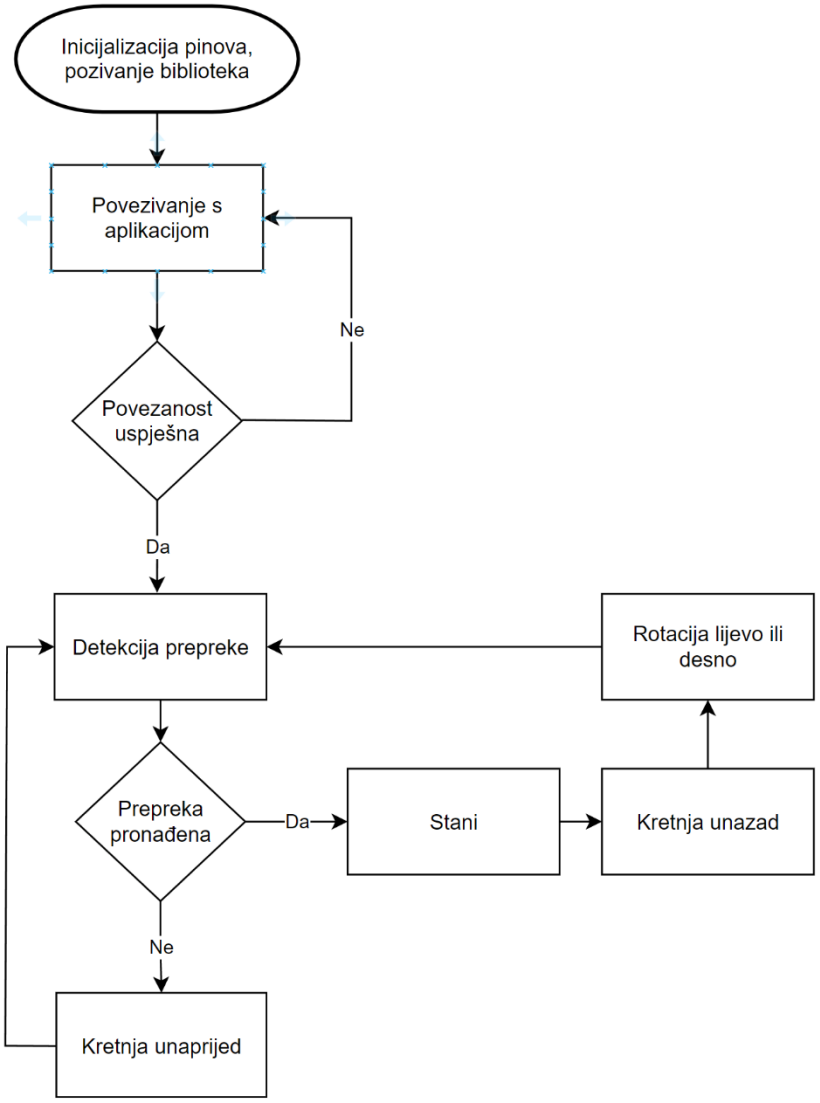
Izvan *loop* dijela programskog koda definirane su funkcije kretanja: *goForward, goForward2, goBackward, stop, goRight, goLeft*. Razlog postojanja dvije funkcije za kretnju u naprijed je taj što je poželjno da se čistač sporije kreće kako bi što bolje očistio površinu ali nastaje problem kada čistač treba krenuti nakon stajanja ili kada se treba popeti na višu površinu kao što je tepih jer se kotači zbog neravnopravnog otpora a i ispražnjenosti baterije neće vrtjeti ujednačeno a može se dogoditi i da se jedan kotač ne počne kretati. Unutra svake funkcije mijenjaju se pinovi za upravljanje smjera vrtnje pogonskih motora, mijenjaju se i dva pina za kontrolu brzine vrtnje. Zatim dolazimo do *loop* dijela programa. Na početku *loop*-a uspostavlja se *bluetooth* konekcija s aplikacijom, s aplikacije čistač prima vrijednosti koje određuju jedan od tri režima rada čistača a to su: kretanje i izbjegavanje prepreka bez mogućnosti detekcije boje, kretanje i izbjegavanje prepreka s mogućnosti detekcije boje

te ručna kontrola kretanja preko aplikacije. Zatim se očitavaju vrijednosti sa senzora boja i ultrazvučnog senzora, ako je udaljenost od prepreke veća od deset centimetara te nema detektirane crvene boje ukoliko je izabran režim rada s detekcijom boja čistač će se početi kretati ravno pomoću brže funkcije za kretanje *goForward* te će nastavi s kretnjom s sporijom funkcijom *goForward2*. Ukoliko se sedam sekundi čistač kreće unaprijed stati će i kretati će se unatrag 100ms. Razlog toga je taj što kada se čistač susretne s kompleksnom preprekom postoji vjerojatnost da ju neće detektirati te će se s njom sudariti i neće ju uspjeti detektirati te će i dalje pozivati funkciju za kretanje naprijed te će na taj način zapravo zapeti, kako korisnik ne bih morao ručno resetirati položaj čistača svakih sedam sekundi poziva se funkcija za kretanje u nazad.

Ukoliko čistač detektira prepreku unutar deset centimetara ili crvenu boju ako je u režimu rada s detekcijom boja onda se izvršavaju sljedeće radnje: poziva se funkcija *stop* zatim funkcija *goBackward* te ponovno funkcija *stop* zatim se nasumično odabire smjer vrtnje pomoću funkcija *goRight* i *goLeft s* vjerojatnostima 50% za lijevo odnosno 50% za desno. Kut rotacija je isto nasumično odabran tako da vrijeme rotacije odnosno vrijeme pozivanja funkcije *goRight* ili *goLeft* iznosi 250ms do 750ms. Razlog toga je što je poželjno imati što nasumičnije kretanje čistača kako bi očistio što veću površinu i kako bi se izbjeglo kretanje u krug. Nakon rotacije čistača ponovno se provjerava udaljenost od prepreke i prisutnost crvene boje i ako nije detektirana prepreka ili crvena boja poziva se funkcija *goForward*.

3.3.1. Prvi režim rada: Kretanje i izbjegavanje prepreke bez mogućnosti detekcije boje

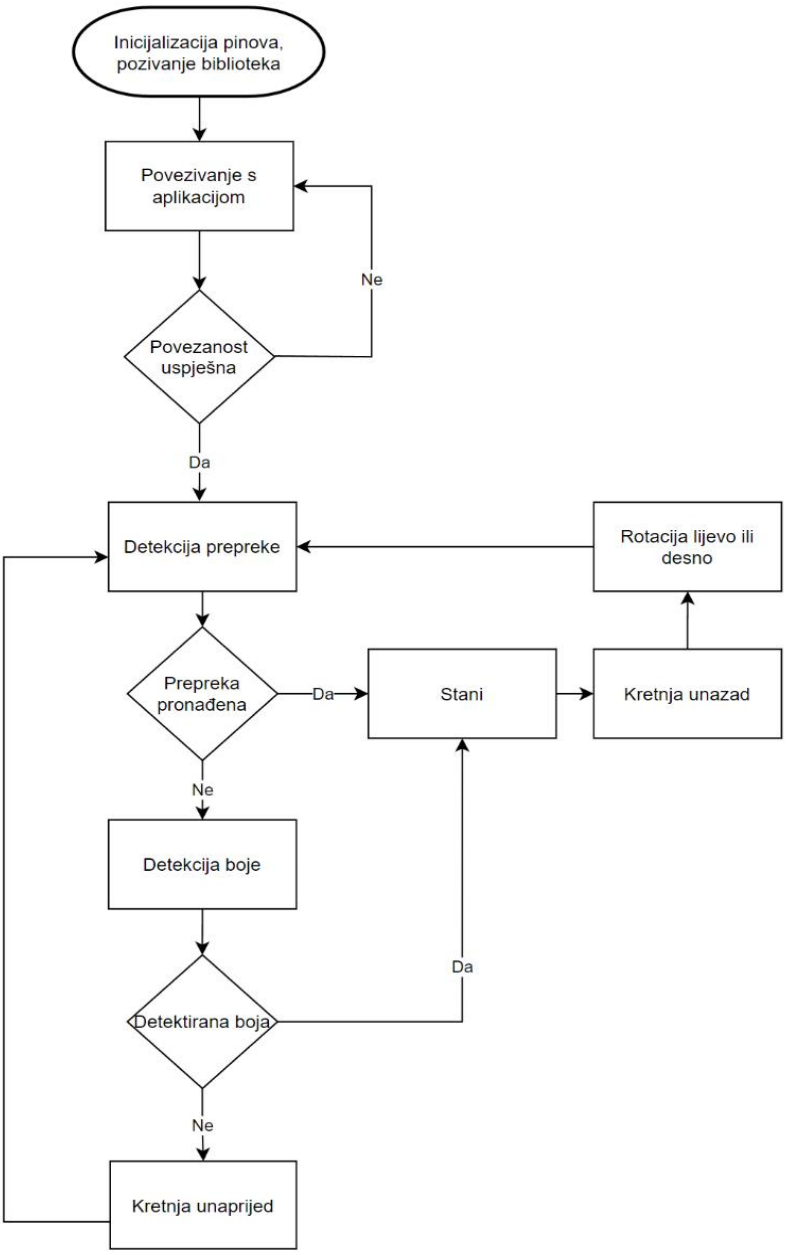
Iz dijagrama na slici 3.9. uočavamo slijed izvršavanja programa. Na početku u *setup* dijeli programa izvršava se inicijalizacija pinova te pozivanje biblioteka zatim dolazimo do *loop* dijela programa gdje se na samome početku provjerava uspješnost povezivanja s aplikacijom pomoću *bluetooth*-a, ukoliko povezivanje nije uspješno čistač neće započeti s kretnjom sve dok se uspješno ne poveže. Nakon uspješnog povezivanja s aplikacijom čistač očitava udaljenost od prepreke pomoću ultrazvučnog senzora te ukoliko detektira prepreku unutar deset centimetara čistač staje, krene unazad, ponovno se zaustavlja te se nasumično rotira u lijevo ili desno na nasumično odabrano vrijeme između 250 i 750 milisekundi zatim ponovno obavlja detekciju prepreke i ukoliko ne pronađe prepreku unutar 10 centimetara počinje se kretati unaprijed.



Slika 3.9. Dijagram toka prvog režima rada.

3.3.2. Drugi režim rada: Kretanje i izbjegavanje prepreke s mogućnosti detekcije boje

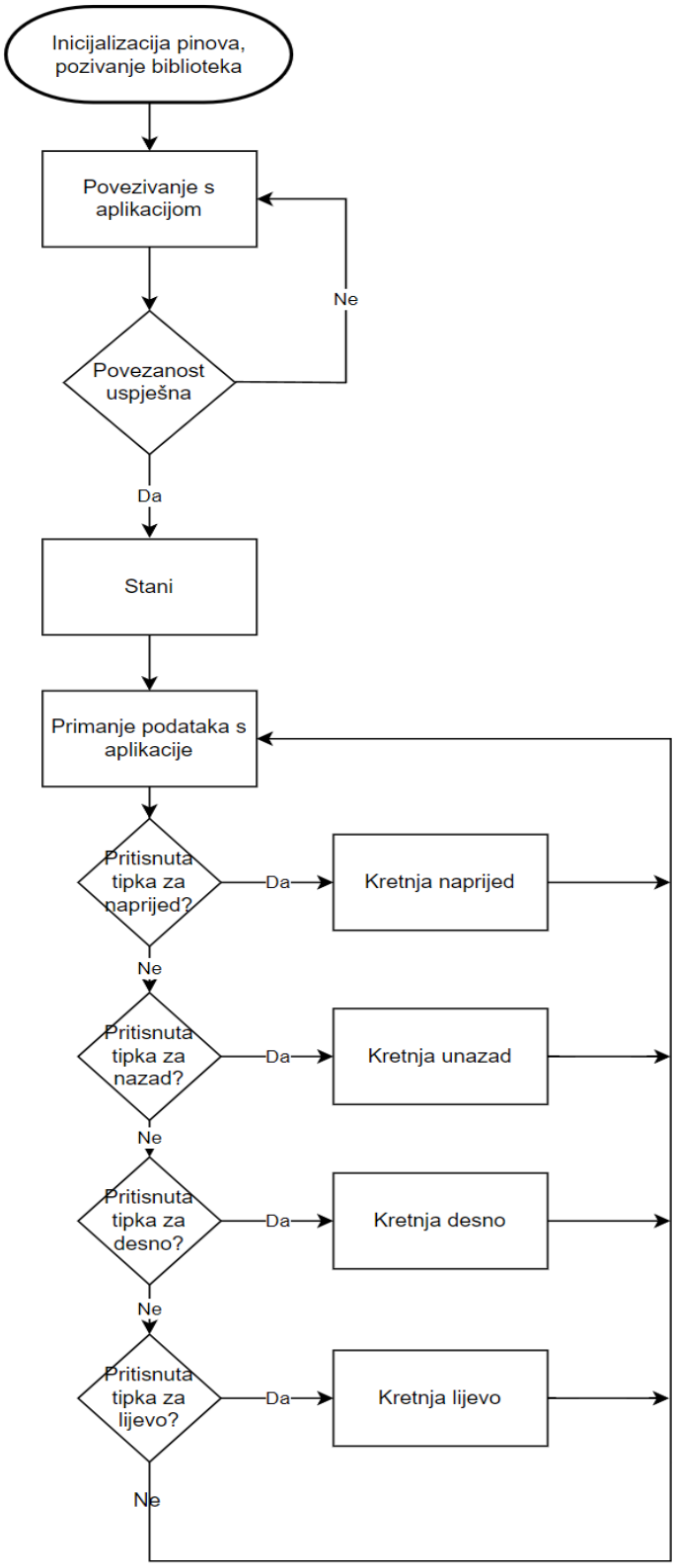
Iz dijagrama na slici 3.10. uočavamo da je redoslijed radnji čistača vrlo sličan kao i u prethodnome režimu rada. Jedina razlika je u tome što nakon detekcije prepreke slijedi očitavanje boje pomoću senzora boja. Ukoliko je očitana definirana crvena boja čistač staje i izvršava radnje kao i kada detektira prepreku.



Slika 3.10. Dijagram toka drugog režima rada.

3.3.3. Treći režim rada: Ručna kontrola kretanja putem aplikacije

U ovom režimu rada kao i u prethodna dva režima čistač obavlja radnje inicijalizacije pinova, pozivanje biblioteka te povezivanje s aplikacijom. Nakon uspješnog povezivanja s aplikacijom pomoću *bluetooth*-a čistač stoji na mjestu sve dok je na aplikaciji pritisnuta neka od tipki za kretanje, nije moguće izvršavati kretnje u dva ili više smjerova u jednom trenutku kao što je vidljivo na slici 3.11.



Slika 3.11. Dijagram toka trećeg režima rada.

4. TESTIRANJE I REZULTATI

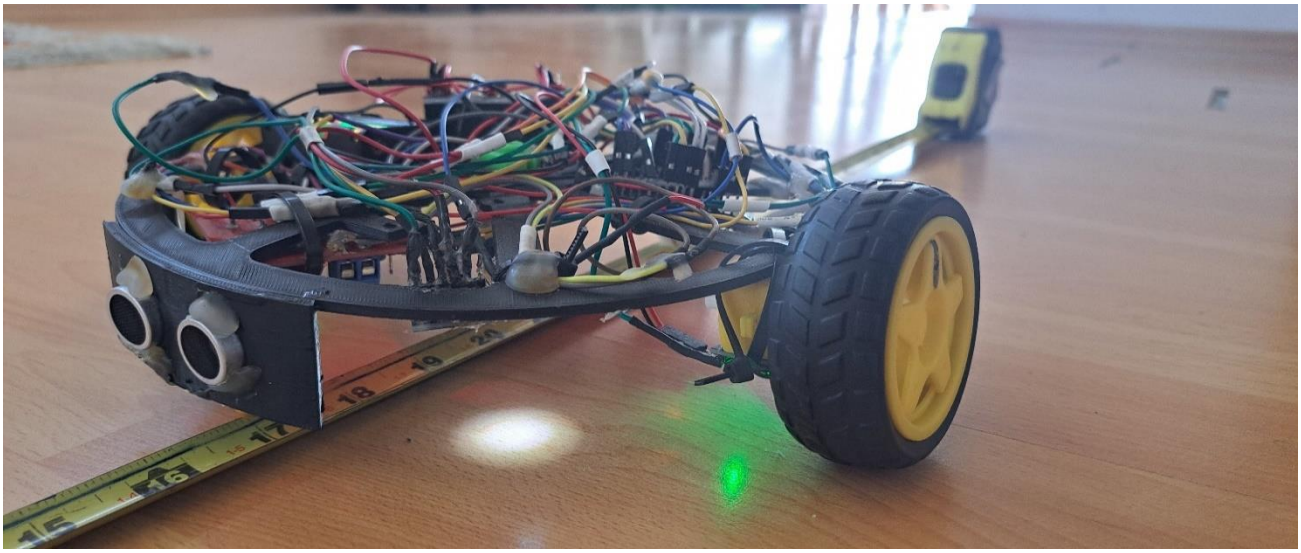
4.1. Metodologija testiranja

Kako bi najbolje optimizirali automatizirani čistač potrebno je provesti određena testiranja. Jedan od najvažnijih parametara optimalnog rada čistača koji treba odrediti je udaljenost na kojoj čistač detektira prepreke, kako bi odredili taj parametar testiranjem čistač se pravocrtno kreće prema prepreci te nakon što se zaustavi izmjeri se udaljenost čistača od prepreke te se najoptimalniji rezultat unosi u programski kod. Potrebno je i odrediti optimalnu debljinu crte u boji koja ograničava zonu kretanja čistača te i sposobnost čistača da u mračnim uvjetima rada detektira crtu u boji.

4.2. Rezultati testiranja

4.2.1. Testiranje udaljenosti zaustavljanja nakon detekcije prepreke s obzirom na unesene parametre

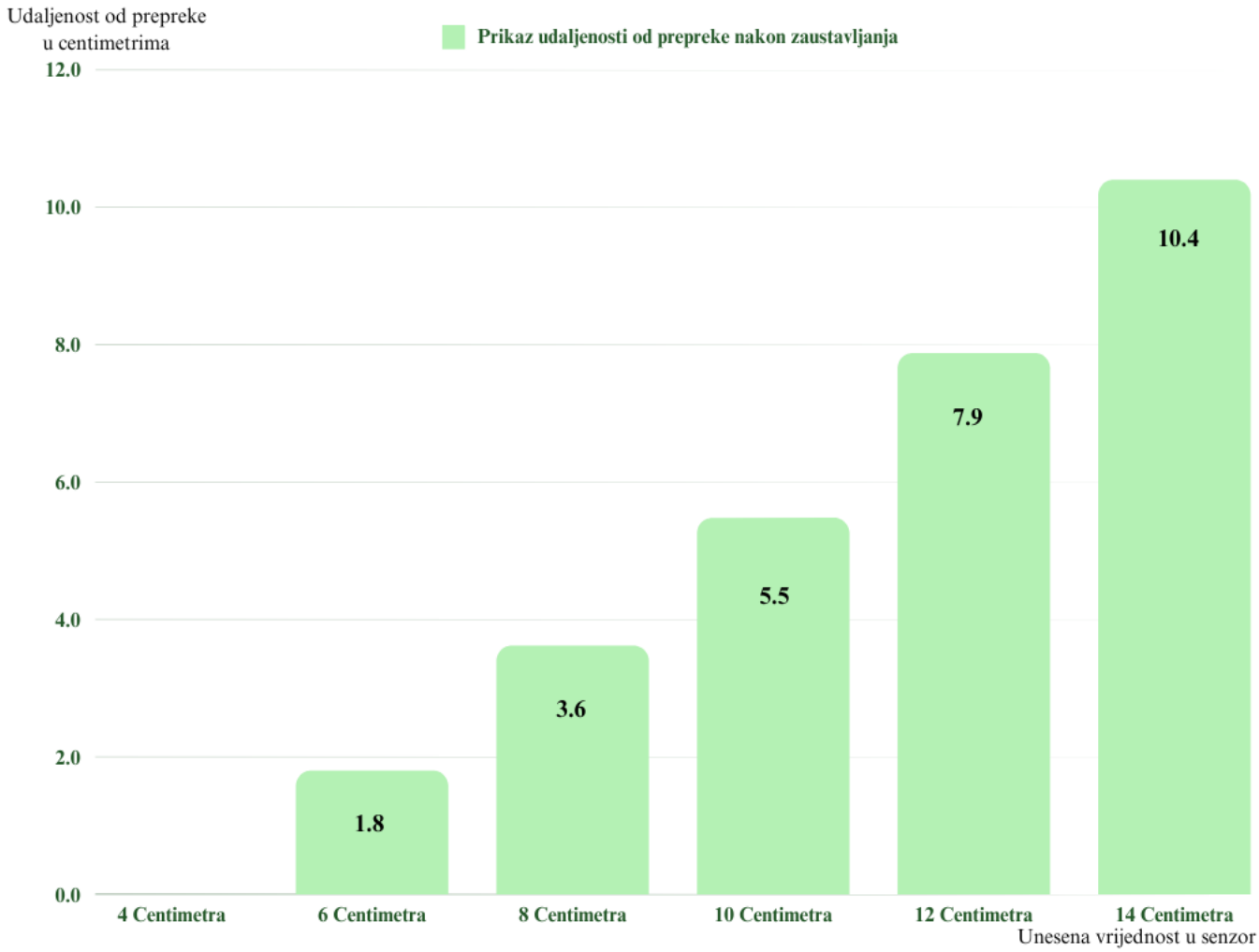
Ovim testiranjem cilj je utvrditi najmanju optimalnu udaljenost na kojoj bi se čistač zaustavio bez da se sudari s preprekom. Testiranje će biti provedeno na način da se čistač postavi okomito od prepreke na dovoljno dalekoj udaljenosti kako bi postigao maksimalnu brzinu kretanja. Čistač se prema prepreci kreće na mjernom instrumentu za udaljenost. Nakon detekcije prepreke čistač se zaustavlja na 5 sekundi kako bi se očitala vrijednost s instrumenta. U programski kod unose se različite veličine na kojima bi se čistač trebao zaustaviti te se vrijednosti na kojima se čistač zapravo zaustavio unose u tablicu 4.1., testiranje za svaku udaljenost se sprovodi 4 puta.



Slika 4.1. Testiranje zaustavne udaljenosti.

Tablica 4.1. Rezultati testiranja zaustavne udaljenosti nakon detektirane prepreke.

	4cm	6cm	8cm	10cm	12cm	14cm
1. Testiranje	0 cm	1.50 cm	3.70 cm	5.0 cm	7.40 cm	10.50 cm
2.Testiranje	0 cm	2.60 cm	4.10 cm	4.60 cm	8.20 cm	10.30 cm
3. Testiranje	0 cm	1.40 cm	3.20 cm	6.30 cm	8.30 cm	11.10 cm
4.Testiranje	0 cm	1.70 cm	3.50 cm	6.0 cm	7.60 cm	9.70 cm
Aritmetička sredina	0cm	1.80 cm	3.62cm	5.48cm	7.88cm	10.40cm



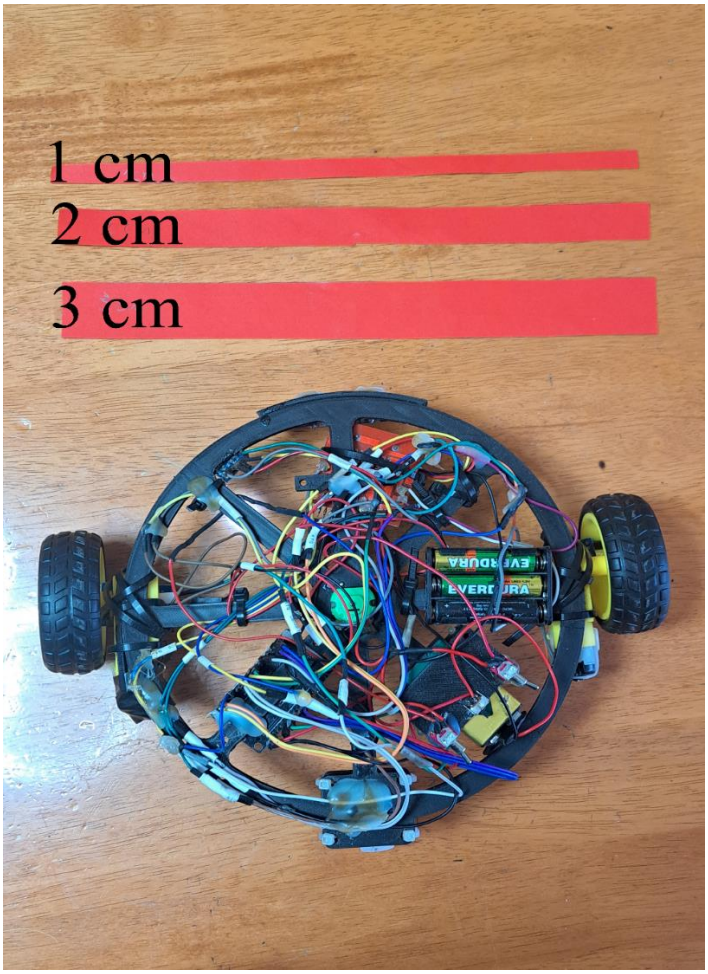
Slika 4.2. Grafički prikaz izmjerenih rezultata u prvom testiranju.

Iz tablice 4.1. uočavamo kako se čistač uspješno zaustavio ispred prepreke za sve unesene vrijednosti osim u slučaju 4 centimetra gdje je došlo do blagog sudara u kojima je čistač uspješno detektirao prepreku ali se nije stigao pravovremeno zaustaviti. Iako se čistač uspješno zaustavio u slučajima gdje je unesena vrijednost 6 i 8 centimetara postoje dva razloga odabira 10 centimetara. Prvi razlog je taj što kada se čistač kreće prema prepreci pod određenim kutom teže očitava udaljenost te često dolazi do sudara s preprekom drugim riječima testiranje je obavljeno u idealnom scenariju gdje se čistač kreće prema prepreci pod 90 stupnjeva i što više kut odstupa od te vrijednosti odnosno što je bliži vrijednostima od 0 stupnjeva i 180 stupnjeva teže će očitati i pravovremeno reagirati na prepreku. Drugi razlog odabira vrijednosti 10 centimetara je taj što se čistač u određenim scenarijima kreće maksimalnom brzinom te se u tim slučajevima ne bi mogao pravovremeno zaustaviti kada bi

mu bile unesene vrijednosti od 6 ili 8 centimetara. Testiranje je sprovedeno s normalnom brzinom kretanja čistača koja je upola manja od njegove maksimalne brzine.

4.2.2. Testiranje različitih debljina crte u boji

Ovim testiranjem cilj je utvrditi najmanju debljinu crte u boji koju čistač može detektirati u osvjetljenim uvjetima rada kao i u mračnim uvjetima rada. Testiranje će biti sprovedeno na način da se čistač kreće prema crti u boji s najvećom brzinom kretanja u osvjetljenim uvjetima rada kao i u mračnim uvjetima rada, u oba uvjeta rada testiranje će biti sprovedeno i s uključenom svjetlećom diodom i s isključenom svjetlećom diodom. Testirati će se 5 različitih debljina crta svaka debljina po 3 puta.

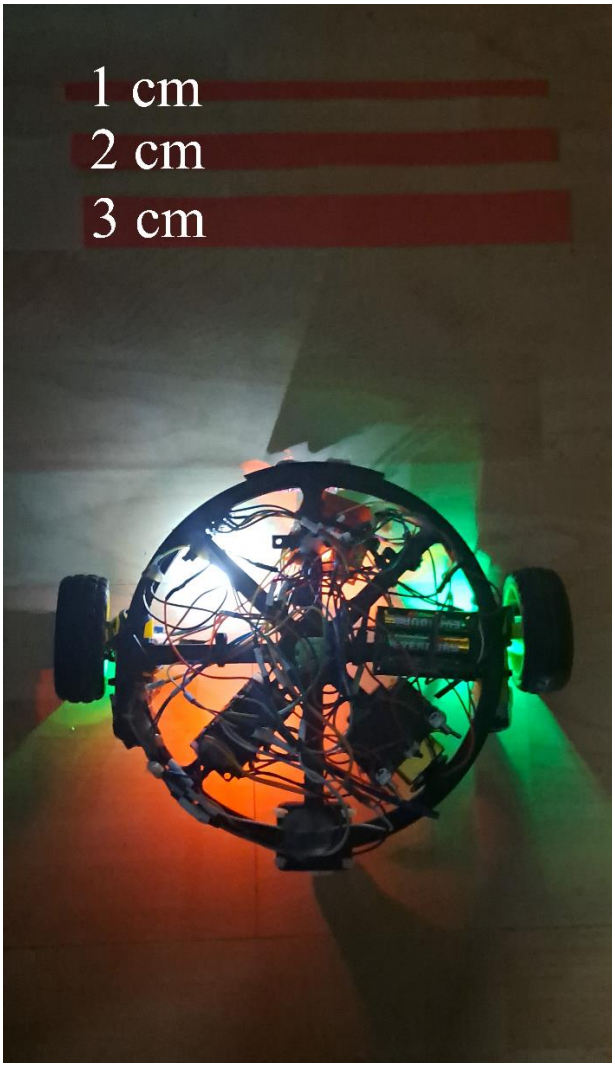


Slika 4.3. Testiranje debljine crte u boji u osvjetljenim uvjetima.

Tablica 4.2. Rezultati testiranja detekcije različitih debljina crta.

Testiranje u osvijetljenoj prostoriji										
	S isključenom svijetlećom diodom					S uključenom svijetlećom diodom				
	1cm	2cm	3cm	4cm	5cm	1cm	2cm	3cm	4cm	5cm
1. Testiranje	-	-	+	+	+	-	-	-	+	+
2. Testiranje	-	-	+	+	+	-	-	-	+	+
3. Testiranje	-	-	+	+	+	-	-	-	+	+

Uspješna detekcija testiranja je označena kao plus a neuspješna detekcija kao minus. Iz tablice 4.2. uočavamo kako je minimalna potrebna debljina crte 4 centimetra ako se koristi svijetleću diodu ili 3 centimetra ako ju se ne koristi. Razlog zašto nisu detektirane crte od 1 i 2 centimetra je taj što je senzor boja podosta odignut od podloge stoga njegov senzor detektira veću kvadratnu površinu nego što je površina crte stoga crvena boja nije dominantna boja i senzor ju ne detektira. Razlog zašto nije detektirana crta od 3 centimetra u testiranju s uključenom svijetlećom diodom je taj što je crvena boja bez korištenja svijetleće diode dominantnija boja u odnosu na zelenu i plavu dok kada se područje ispod senzora boja osvijetli s svijetlećom diodom crvena boja je i dalje dominantnija od zelene i plave ali ne toliko kao u slučaju s isključenom svijetlećom diodom.



Slika 4.4. Testiranje debljine crte u mračnim uvjetima.

Tablica 4.3. Rezultati testiranja u mračnoj prostoriji.

Testiranje u mračnoj prostoriji										
	S isključenom svijetlećom diodom					S uključenom svijetlećom diodom				
	1cm	2cm	3cm	4cm	5cm	1cm	2cm	3cm	4cm	5cm
1. Testiranje	-	-	-	-	-	-	-	-	+	+
2. Testiranje	-	-	-	-	-	-	-	-	+	+
3. Testiranje	-	-	-	-	-	-	-	-	+	+

Prilikom testiranja u mračnoj prostoriji s isključenom svijetlećom diodom vidljivom na slici 4.4. iz tablice 4.3 uočavamo kako u niti jednom testiranju nije detektirana crta crvene boje dok su u testiranju s uključenom svijetlećom diodom uspješno detektirane crte debljine 4 i 5 centimetara u sva tri testiranja te se rezultati podudaraju s testiranjem u osvijetljenoj prostoriji kada je uključena svijetleća dioda iz toga možemo zaključiti da korištena svijetleća dioda savršeno obavlja svoj zadatak.

5. ZAKLJUČAK

Mobilni čistač podova mali je kućanski robot koji postoji kako bi ljudima olakšao život tako da ih oslobodi od obaveze čišćenja podova. Taj posao obavlja pomoću raznih senzora i aktuatora i unaprijed smišljenih algoritama. Koriste se razni senzori koji su neophodni u kretnji samoga čistača, oni mogu prepoznavati razne prepreke od onih jednostavnih kao što su zidovi do onih kompleksnijih kao što su kablovi ili odjeća na podu, imaju i mogućnost prepoznavanja padova odnosno stepenica. Ti senzori šalju podatke o nadolazećim preprekama mikroupravljaču koji zatim odlučuje na temelju algoritama kretanja kako dalje postupiti.

Ovaj rad je predstavljao četiri izazova: izrada same konstrukcije čistača, detekcija prepreka i reakcija na njih, prepoznavanje podnih crta u boji te reakcija na njih i mogućnost upravljanja čistačem pomoću mobilne aplikacije.

Čistač je izrađen od 3D printane plastike, debljina kostura čistača je 3 milimetra te se zbog težine samih komponenti te mnogobrojnih sudara prilikom testiranja kostur čistaća se izvitoperio te bi bilo poželjno da je kostur veće debljine. Promjer čistača iznosi 20 centimetara i to je predstavljalo problem prilikom raspodijele komponenti te se stvarni raspored komponenti poprilično razlikuje od onog prvobitno zamišljenog u drugom poglavlju iz istog razloga koristi se rotacijska četkica umjesto rotacijskog valjka kako je prvobitno zamišljeno. Još jedna razlika je i u izvoru napajanja, motor metlice se napaja iz 3 AAA baterije od 1.5 volta serijski spojenih kako bi dobili 4.5 volta kako ne bi glavni izvor napajanja bio preopterećen. Problem je predstavljalo i to što se čistač ne kreće ravno zbog toga što jedan kotač pruža veći otpor zakretanja taj problem je riješen unutar koda tako što dobiva veći napon pomoću *PWM*-a od drugog motora.

Detekcija prepreka postignuta je korištenjem ultrazvučnog senzora, jedini problem u ovom dijelu predstavljala je brzina detekcije i reakcije na prepreku što je riješeno unutar koda tako da se što manje koristi *delay* funkcija kako bi se program što brže izvršavao. Bilo je potrebno i pronaći optimalnu vrijednost na kojoj bi se čistač trebao zaustaviti prilikom detekcije prepreke, kroz testiranje je izabrana vrijednost od 10 centimetara. Može se dogoditi da se zbog kompleksnosti prepreke čistač sudari i ne detektira prepreku te se i dalje poziva funkcija za kretanje u naprijed odnosno kotači mu se vrte u mjestu zbog toga se nakon 7 sekundi neprekidnog kretanja u naprijed poziva funkcija za kretanje u nazad u trajanju od 100 milisekundi kako bi se čistač odvoji od prepreke s kojom se sudario. Zbog ispražnjenosti baterije te različitog otpora koju podloga pruža kotačima čistača moguće je da se

prilikom početka kretanja nakon stajanja jedan kotač a ponekad i oba ne okreću. To je riješeno tako da se na početku kretanja poziva funkcija *goForward* koja čistaču omogućuje kretanje maksimalnom brzinom odnosno na motore se šalje najveći mogući napon pomoću *PWM*-a kako bi osigurali da oba kotača savladaju otpor površine.

Prepoznavanje podne crte postignuto je korištenjem senzora boja te je izabrana boja u dogovoru s mentorom crvena. Ovdje se pojavio problem detekcije boje u mračnijim uvjetima rada što je riješeno dodavanjem svijetleće bijele diode od 5 milimetara. Još jedan problem je bio uspješno definirati crvenu boju. Bilo je potrebno i odrediti minimalnu debljinu crte u boji, kroz testiranje izabrana je minimalna debljina od 4 centimetra.

Za izradu mobilne aplikacije koristi se *MIT App Inventor* koji je izabran zbog svoje jednostavnosti korištenja. Aplikacija komunicira s čistačem pomoću *Bluetootha*-a. Postoje tri režima rada koja korisnik može odabrati unutar aplikacije: kretanje s detekcijom prepreke bez detekcije boja, kretanje s detekcijom prepreke s mogućnosti detekcije boja te mogućnost ručnog upravljanja kretnjom čistača u stvarnom vremenu.

LITERATURA

[1] Robotic vacuum cleaner, https://en.wikipedia.org/wiki/Robotic_vacuum_cleaner#:~:text=A%20robotic%20vacuum%20cleaner%2C%20sometimes,programmable%20controllers%20and%20cleaning%20routines.., Pristup:26.6.2023.

[2] MathWorks, https://www.mathworks.com/help/vision/visual-simultaneous-localization-and-mapping-slam.html., Pristup: 26.6.2023.

[3] MIT App Inventor, https://appinventor.mit.edu/about-us., Pristup:30.6.2023.

[4] MIT App Inventor, https://appinventor.mit.edu/explore/library., Pristup: 28.6.2023.

[5] HCSR04 Datasheet, https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf., Pristup: 20.8.2024.

[6] L298N Motor Driver, https://components101.com/sites/default/files/component_datasheet/L298N-Motor-Driver-Datasheet.pdf., Pristup: 20.8.2024.

[7] APDS 9960 Datasheet, https://cdn.sparkfun.com/assets/learn_tutorials/3/2/1/Avago-APDS-9960-datasheet.pdf.,Pristup: 20.8.2024.

[8] ESP32 Wroom datasheet, https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf., Pristup: 20.8.2024.

[9] Nickel Metal Hydride Batteries datasheet, https://www.masterinstruments.com.au/files/data_sheets/NiMh%20Cyl%20&%20Prism/Panasonic%20NiMH%20Catalogue.pdf., Pristup: 20.8.2024.

SAŽETAK

Zadatak ovog završnog rada bio je projektirati i dizajnirati automatizirani čistač podova sa sustavom detekcije zone kretanja korištenjem crte u boji s mogućnošću kontrole mobilnom aplikacijom. Čistač treba imati mogućnost detekcije prepreka i izbjegavati iste te mora imati i mogućnost detekcije boje. Logika čistača izvedena je na mikroupravljaču ESP32 WROOM koji je programiran u razvojnom okruženju Arduino IDE a mobilna aplikacija izrađena je pomoću besplatne platforme *MIT App Inventor*.

Ključne riječi: automatizirani čistač podova, Android aplikacija, Arduino IDE, ESP32 WROOM, mikroupravljač

ABSTRACT

AUTOMATED FLOOR CLEANER WITH MOTION ZONE DETECTION SYSTEM BASED ON COLOR ANALYSIS

The task of this final paper was to design an automated floor cleaner with a motion zone detection system using a colored line with the possibility of control with a mobile application. The cleaner should have the ability to detect obstacles and avoid them, and it must also have the ability to detect color. The logic of the cleaner is implemented on the ESP32 WROOM microcontroller, which is programmed in the Arduino IDE development environment, and the mobile application was created using the free MIT App Inventor platform.

Keywords: automated floor cleaner, Android application, Arduino IDE, ESP32 WROOM, microcontroller

ŽIVOTOPIS

Filip Murić rođen 23.9.1999. godine u Osijeku. Pohađao Osnovnu školu Bilje. Obrazovanje nastavlja u Elektrotehničkoj i prometnoj školi Osijek 2014. godine, smjer tehničar za elektroniku. Nakon toga upisuje stručni studij elektrotehnike smjer automatika na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

PRILOZI I DODACI

Hardverske komponente

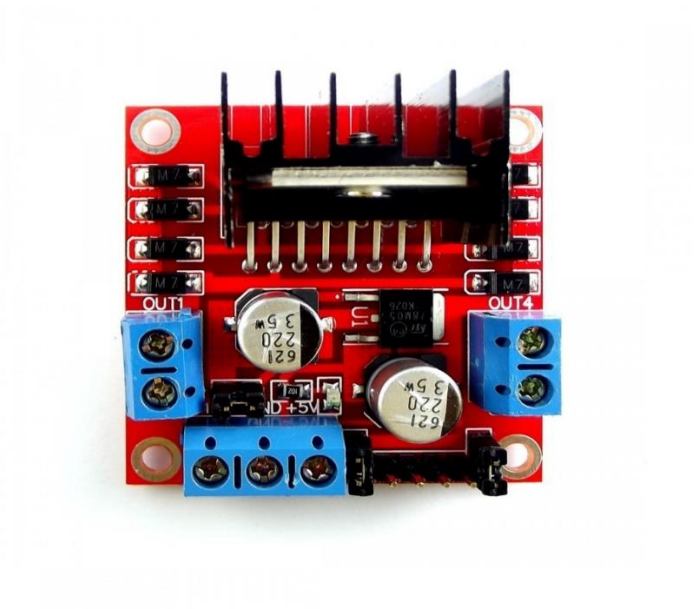
P.1. - Ultrazvučni senzor HC-SR04

Ultrazvučni senzor pruža mogućnost beskontaktnog mjerenja udaljenosti s preciznosti mjerenja od 3 mm. Funkcionira na način da preko *I/O* pinova šalje signal najmanje 10 μs, šalje osam pulsnih signala od 40 kHz koji se odbiju nazad u senzor ako se ispred senzora nalazi prepreka, na temelju tih signala moguće je izračunati udaljenost. Senzor se napaja s 5V te radi na struji od 15mA. Osim dva pina za napajanje i uzemljenje ima još dva pina *Echo* i *Trig* pomoću kojih se šalju odnosno primaju signali. Predviđeni raspon detekcije je od 2 centimetra do 400 centimetara **[5]**.

Slika 7.1. Ultrazvučni senzor HC-SR04

P.2. - Motor *driver* L298N

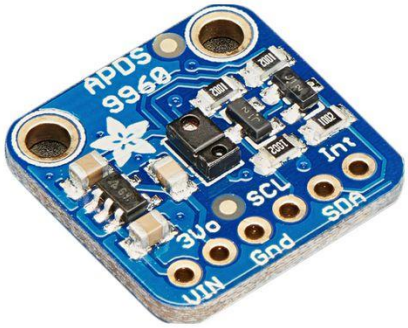
Elektronički sklop(dvostruki H most) koji omogućava upravljanje dva istosmjerna motora odnosno omogućuje kontroliranje brzine vrtnje motora te smjer vrtnje motora. Motor *driver* ima dva bloka za spajanje motora 1 i motora 2 te još jedan blok za *Vcc*, uzemljenje te pin 5V. Moguća su dva režima rada u prvome ako se na *Vcc* dovede od 5 volta do 12 volta onda se izlaz 5V može koristiti kao izvor napajanja za mikroupravljač a ukoliko se na *Vcc* dovede između 12 i 35 volta onda je potrebno na pin 5V dovesti 5 volta. Maksimalna struja je 2A **[6]**.



Slika 7.2. Motor *driver* L298N

P.3. - Senzor boja APDS9960

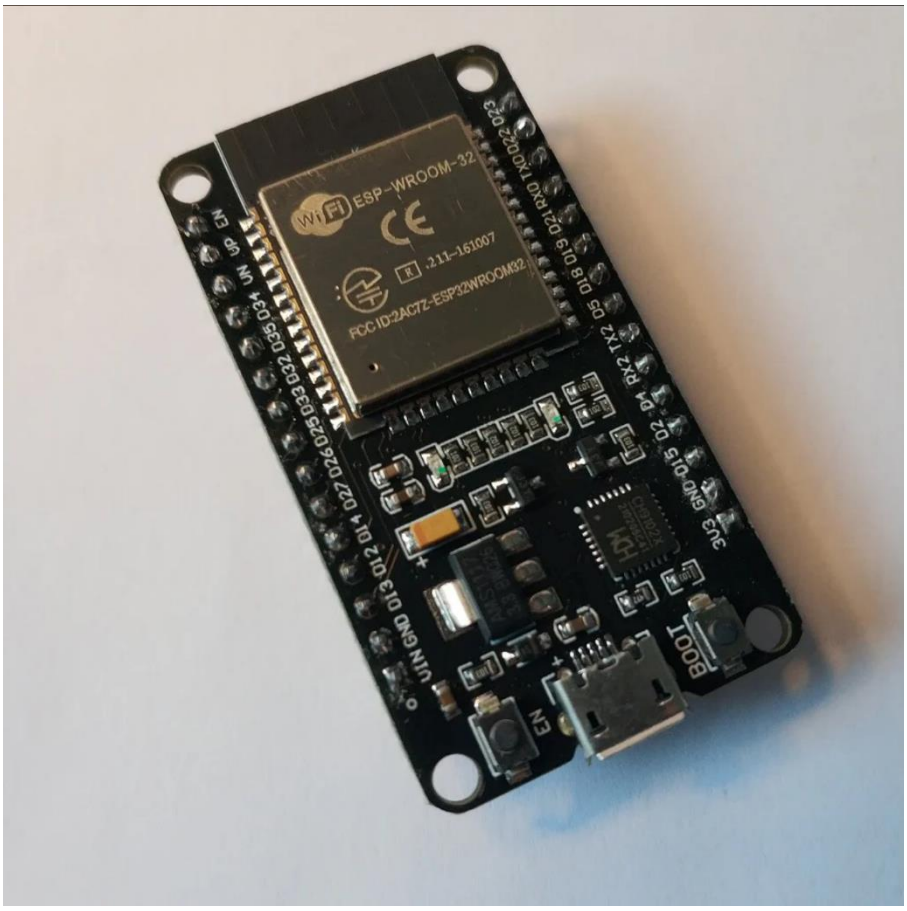
Elektronički sklop s mogućnostima prepoznavanja boja, ambijentalnog osvjetljenja, detekcijom blizine i detekcijom gestikulacija. Napaja se s maksimalno 3.8 Volta, posjeduje dva *I/O* pina koji se koriste za I^2C komunikaciju, pin za napajanje te pin za uzemljenje. Mogućnost detekcije *RGB* boja te ambijentalnog osvjetljenja **[7]**.



Slika 7.3. Senzor boja APDS9960

P.4. - Mikroupravljač ESP32-WROOM

Modul s širokim rasponom primjena od upravljanja senzorima na niskim naponima do zahtjevnih primjena kao što su kodiranje glasa te dekodiranje MP3. Ima mogućnost rada preko WI-FI mreže, *Bluetooth*-a te *Bluetooth* LE. Koristi ESP32-D0WDQ6 čip, ima dvije procesorske jezgre koje imaju podesiv takt frekvencije između 80 MHz do 240MHz. Čip posjeduje i koprocesor male snage kako bi efikasnije obavljao zadatke koji nisu zahtjevni **[8]**.



Slika 7.4. Mikroupravljač ESP32 Wroom

P.5. - Izvori napajanja

Za napajanje glavnog sklopovlja koristi se punjiva baterija od 9 volta kapaciteta 200mAh na bazi nikla tipa Ni-MH(Nikal-metal-hidrid). Ovaj tip baterije nastao je kao zamjena za nikal kadmij baterije

zbog toga što je kadmij toksičan. Kadmij je zamijenjen ne toksičnim hidridom. Prednosti ovog tipa nad nikal kadmij baterijom je taj što je gustoća energije 50% veća odnosno kapacitet im je od 2 do 3 puta veći. Mane ovog tipa baterije u odnosu na nikal kadmij baterije su dvostruko kraći životni vijek te snažnije samopražnjenje baterije koje je najveće u prvom danu nakon punjenja a s vremenom taj efekt opada **[9]**.

Slika 7.5. Punjiva baterija od 9V

Za napajanje motora četkice čistača koriste se tri ne punjive alkalne AAA 1.5 Voltne baterije spojene serijski. Vrlo popularne baterije koje se mogu pronaći u kućanstvima od daljinskih upravljača, digitalnih kamera, igračkama te mnogim drugim elektroničkim napravama. Naziv alkalna baterija proizlazi iz toga što je elektrolit lužina odnosno kalijev hidroksid. Katoda je izrađena od manganovog dioksida a anoda od cinka u prahu. Iako više ne sadrže otrovne tvari poput žive i kadmija i dalje mogu biti opasne po okoliš i zdravlje zbog toga što su sklone curenju kalijevog hidroksida koji može izazvati iritaciju kože očiju te dišnih puteva.

Slika 7.6. Baterije motora četkice

P.6. - Istosmjerni motor

Ovaj tip motora koristi istosmjernu struju kako bi proizveo mehaničku energiju. Radi na rasponu od 3 do 12 volta te maksimalnoj struji od 250 mA. Posjeduje zupčanike kako bi minimizirao gubitak energije i omogućio preciznije upravljanje brzinom vrtnje kotača. Posjeduje *EMC* (Elektromagnetsku kompatibilnost) koja neutralizira elektromagnetske smetnje zbog kojih bi pri većem opterećenju motora došlo do prevelikih smetnji koji bi poremetile normalan rad mikroupravljača.

Slika 7.7. Pogonski motori i kotači

Programski kod

```
#include <SparkFun_APDS9960.h>
#include <Wire.h>
#include <BluetoothSerial.h>


BluetoothSerial ESP_BT;
int incoming;
int stani_kreni;
int boja;
int manual_control;
int naprijed, nazad, desno, ljevo;

//UZS (Definicija pinova i deklariranje varijabli za ultrazvučni senzor)
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;

//COLORSENSOR (Deklaracija varijabli za senzor boja)
SparkFun_APDS9960 apds = SparkFun_APDS9960();
uint16_t ambient_light = 0;
uint16_t red_light = 0;
uint16_t green_light = 0;
uint16_t blue_light = 0;
int red = 0;

//PWM (Definicija pinova i deklaracija varijabli za upravljanje brzinom)
int speed1, speed2, speed3;
int ena = 14;
int enb = 13;

```

Slika 7.8. Prikaz pozivanja biblioteka, definiranja varijabli i konstanti.

```
// Ljevi motor
int leftMotorForwardPin = 17;
int leftMotorBackwardPin = 16;

// Desni motor
int rightMotorForwardPin = 32;
int rightMotorBackwardPin = 33;

// Pomocne varijable za timere
unsigned long currentMillis = 0;
unsigned long currentMillis2 = 0;
unsigned long previousMillis = 0;
unsigned long previousMillis2 = 0;
unsigned long interval = 7000;
unsigned long interval2 = 3000;

void setup() {
    pinMode(leftMotorForwardPin, OUTPUT);
    pinMode(leftMotorBackwardPin, OUTPUT);
    pinMode(rightMotorForwardPin, OUTPUT);
    pinMode(rightMotorBackwardPin, OUTPUT);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);

    Serial.begin(115200);

    ESP_BT.begin("ESP32_CONTROL");


   // Inicijalizacija APDS-9960 senzora boja
  if ( apds.init() ) {
    Serial.println(F("APDS-9960 initialization complete"));
  } else {
    Serial.println(F("Something went wrong during APDS-9960 init!"));
  }
  if ( apds.enableLightSensor(false) ) {
    Serial.println(F("Light sensor is now running"));
  } else {
    Serial.println(F("Something went wrong during light sensor init!"));
  }
}

```

Slika 7.9. Definiranje pinova, setup dio koda.

```
75  void loop() {
76    // Povezivanje s aplikacijom te primanja podataka putem bluetooth-a
77    if (ESP_BT.available()){
78      incoming = ESP_BT.read();
79      int button = floor(incoming/10);
80      int value = incoming % 10;
81      Serial.print("SALJE SE:   "), Serial.println(incoming);
82
83      // Logika odabranog režima rada
84      switch(button){
85        case 1:
86          Serial.print("Stani/kreni: "), Serial.println(value);
87          stani_kreni = value;
88          break;
89        case 2:
90          Serial.print("Boja: "), Serial.println(value);
91          boja = value;
92          Serial.print("boja: "), Serial.println(boja);
93          break;
94        case 3:
95          Serial.print("Manual Control: "), Serial.println(value);
96          manual_control = value;
97          break;
98        case 4:
99          Serial.print("Naprijed: "), Serial.println(value);
100         naprijed = value;
101         break;
102       case 5:
103         Serial.print("Nazad: "), Serial.println(value);
104         nazad = value;
105         break;
106       case 6:
107         Serial.print("Desno: "), Serial.println(value);
108         desno = value;
109         break;
110       case 7:
111         Serial.print("Ljevo: "), Serial.println(value);
112         ljevo = value;
113         break;
114     }
115   }
```

Slika 7.10. Početak loop dijela programa, primanje podataka s aplikacije.

```
116   //Uvijet za detekciju crvene boje
117   if (((red_light > green_light*2 && red_light > blue_light*2 &&
118   blue_light < green_light*1.2 && blue_light > green_light*0.8)
119   || (red_light > green_light*4 && red_light > blue_light*4)) && boja == 1){
120    red = 1;
121   }
122   else{
123     red = 0;
124   }
125
126   speed1 = 107;
127   speed2 = 100;
128   speed3 = 255;
129   digitalWrite(trigPin, LOW);
130   delayMicroseconds(2);
131   digitalWrite(trigPin, HIGH);
132   delayMicroseconds(10);
133   digitalWrite(trigPin, LOW);
134
135     //Očitanje echo Pina, vraća vrijeme u mikrosekundama
136   duration = pulseIn(echoPin, HIGH);
137
138   // Izračun udaljenosti
139   distance = duration * SOUND_SPEED/2;
140
```

Slika 7.11. Nastav loop dijela programa

```
140   //Logika detekcija prepreka i ili crvene boje
141   if (stani_kreni == 1 && manual_control == 0){
142     if (distance > -1 && distance < 10 || red == 1) {
143       if (red == 1){
144         Serial.println("RED");
145       }
146       Serial.println(distance);
147       stop();
148       delay(500);
149       goBackward();
150       delay(500);
151       stop();
152       delay(500);
153
154       if(random(0, 2) == 0) {
155         goLeft();
156       }
157       else {
158         goRight();
159       }
160
161     delay(250 + (rand()%500));
162     stop();
163     delay(500);
164     previousMillis2 = currentMillis2;
165
166
167     }
```

Slika 7.12. Nastavak loop dijela programa, logika detekcije prepreke.

```
168       else {
169         // Kretanje u nazad nakon 7 neprestanih sekundi kretanja unaprijed
170         currentMillis = millis();
171         if (currentMillis - previousMillis >= interval) {
172           goBackward();
173           delay(100);
174 
175           previousMillis = currentMillis;
176         }
177         else{
178           currentMillis2 = millis();
179           if(currentMillis2 - previousMillis2 >= interval2){
180             goForward2();
181           }
182           else{
183             goForward();
184             delay(100);
185           }
186         }
187       }
188       }
189       //Logika kretanja u ručnom režimu rada
190       if(stani_kreni == 1 && manual_control == 1){
191         if(naprijed == 1 && nazad == 0 && desno == 0 && ljevo == 0){
192           goForward();
193         }
194         if(naprijed == 0 && nazad == 1 && desno == 0 && ljevo == 0){
195           goBackward();
196         }
197         if(naprijed == 0 && nazad == 0 && desno == 1 && ljevo == 0){
198           goLeft();
199         }
200         if(naprijed == 0 && nazad == 0 && desno == 0 && ljevo == 1){
201           goRight();
202         }
203         if(naprijed == 0 && nazad == 0 && desno == 0 && ljevo == 0){
204           stop();
205         }
206       }
```

Slika 7.13. Nastavak loop dijela programa, logika kretanja u ručnom režimu rada.

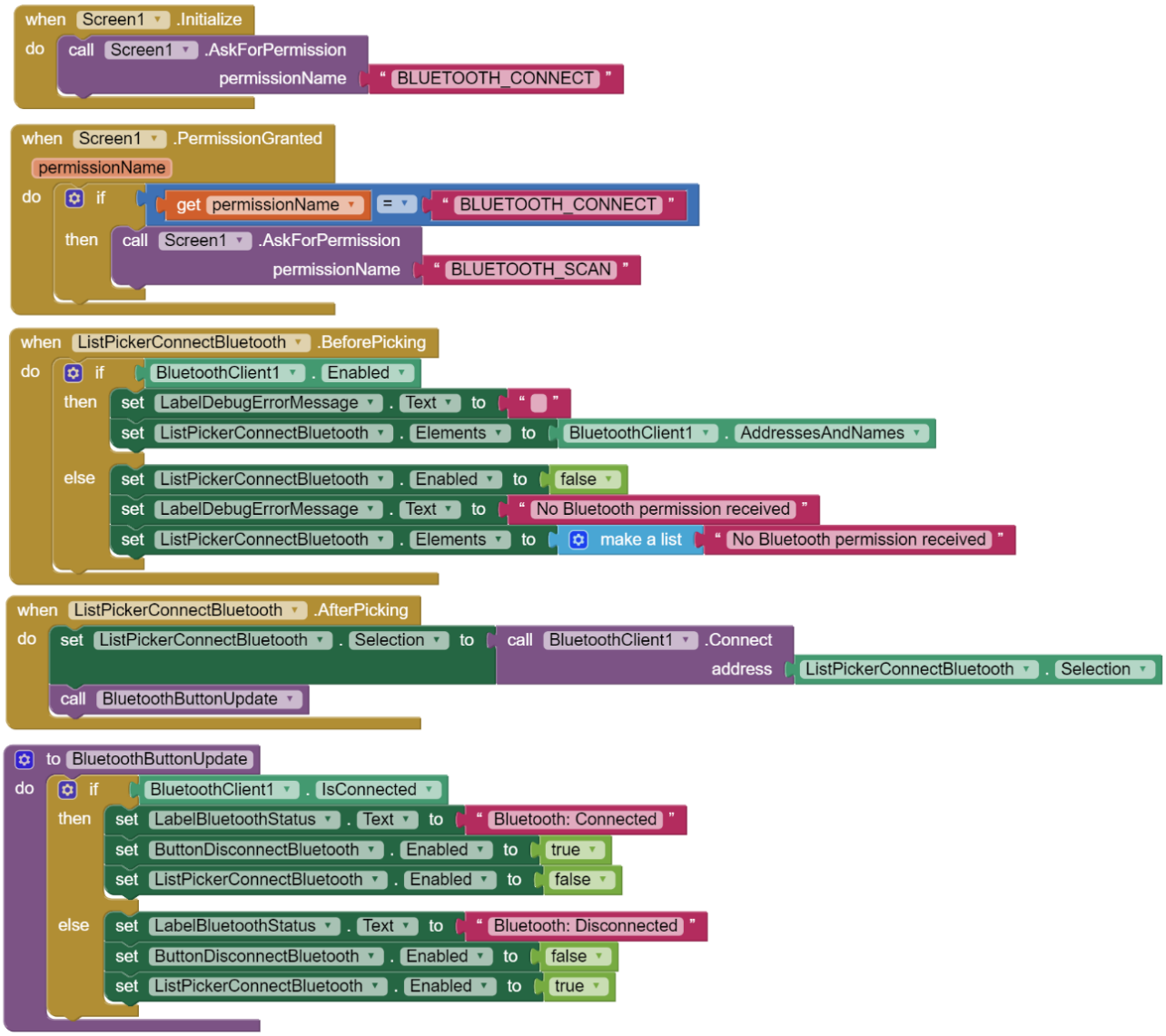
```
207     if(stani_kreni == 0){
208       stop();
209       delay(100);
210     }
211   }
212
213   // Funkcije kretanja
214   void goForward() {
215     digitalWrite(leftMotorForwardPin, HIGH);
216     digitalWrite(leftMotorBackwardPin, LOW);
217     digitalWrite(rightMotorForwardPin, HIGH);
218     digitalWrite(rightMotorBackwardPin, LOW);
219     analogWrite(ena, speed3);
220     analogWrite(enb, speed3);
221   }
222   void goForward2() {
223     digitalWrite(leftMotorForwardPin, HIGH);
224     digitalWrite(leftMotorBackwardPin, LOW);
225     digitalWrite(rightMotorForwardPin, HIGH);
226     digitalWrite(rightMotorBackwardPin, LOW);
227     analogWrite(ena, speed1);
228     analogWrite(enb, speed2);
229   }
230
231   void goBackward() {
232     digitalWrite(leftMotorForwardPin, LOW);
233     digitalWrite(leftMotorBackwardPin, HIGH);
234     digitalWrite(rightMotorForwardPin, LOW);
235     digitalWrite(rightMotorBackwardPin, HIGH);
236     analogWrite(ena, speed3);
237     analogWrite(enb, speed3);
238   }
239
240   void stop() {
241     digitalWrite(leftMotorForwardPin, LOW);
242     digitalWrite(leftMotorBackwardPin, LOW);
243     digitalWrite(rightMotorForwardPin, LOW);
244     digitalWrite(rightMotorBackwardPin, LOW);
245   }
246
```

Slika 7.14. Završetak loop dijela programa, funkcije za kretanje.

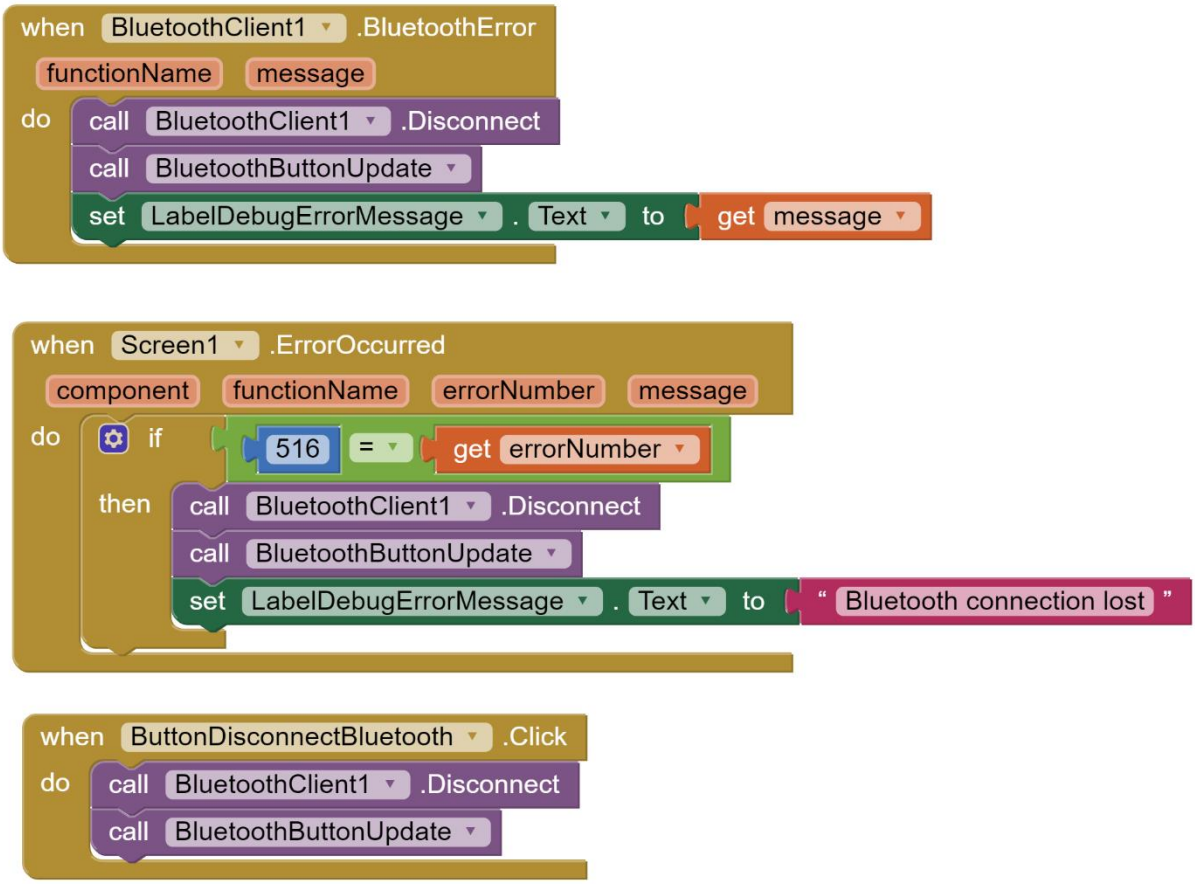
```
246
247 void goRight() {
248   digitalWrite(leftMotorForwardPin, HIGH);
249   digitalWrite(leftMotorBackwardPin, LOW);
250   digitalWrite(rightMotorForwardPin, LOW);
251   digitalWrite(rightMotorBackwardPin, LOW);
252   analogWrite(ena, speed3);
253   analogWrite(enb, speed3);
254 }
255
256 void goLeft() {
257   digitalWrite(leftMotorForwardPin, LOW);
258   digitalWrite(leftMotorBackwardPin, LOW);
259   digitalWrite(rightMotorForwardPin, HIGH);
260   digitalWrite(rightMotorBackwardPin, LOW);
261   analogWrite(ena, speed3);
262   analogWrite(enb, speed3);
263 }
```

Slika 7.15. Funkcije za kretanje.

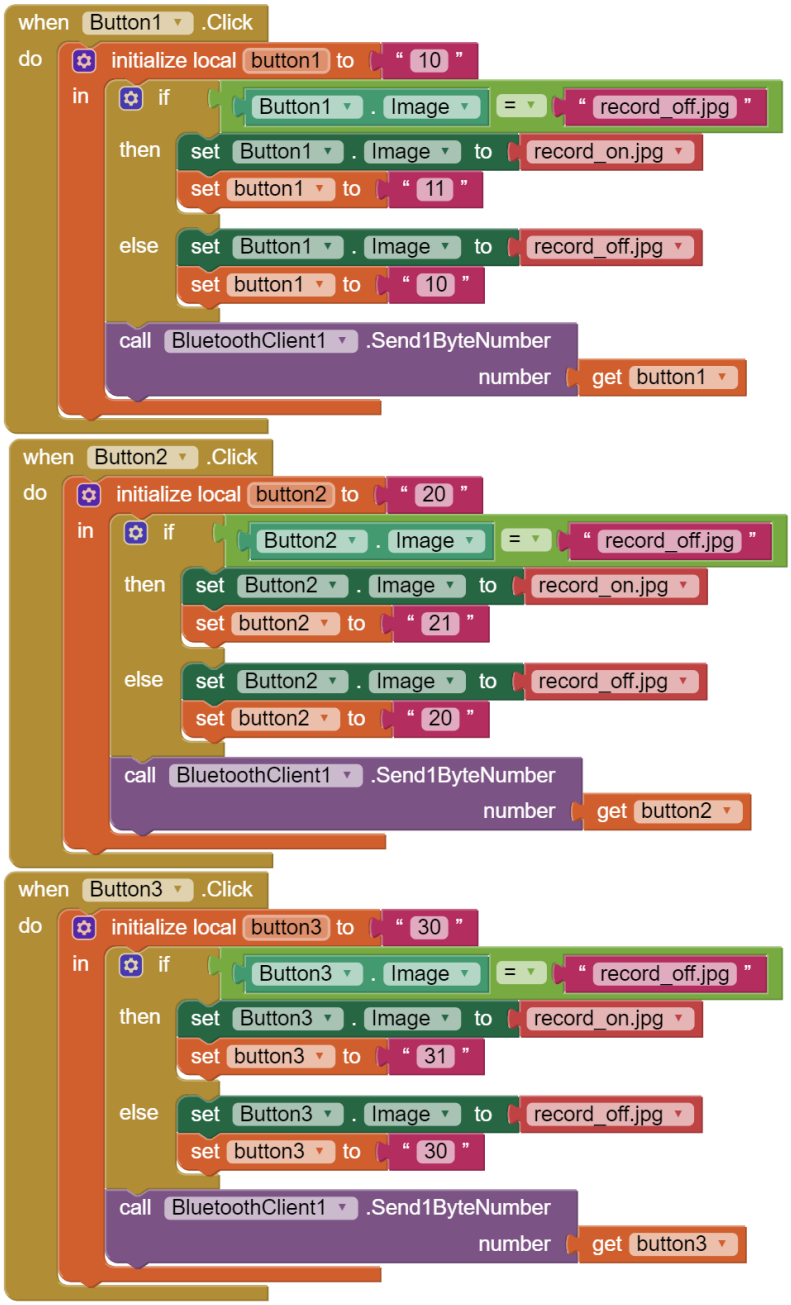
Blokovski kod aplikacije



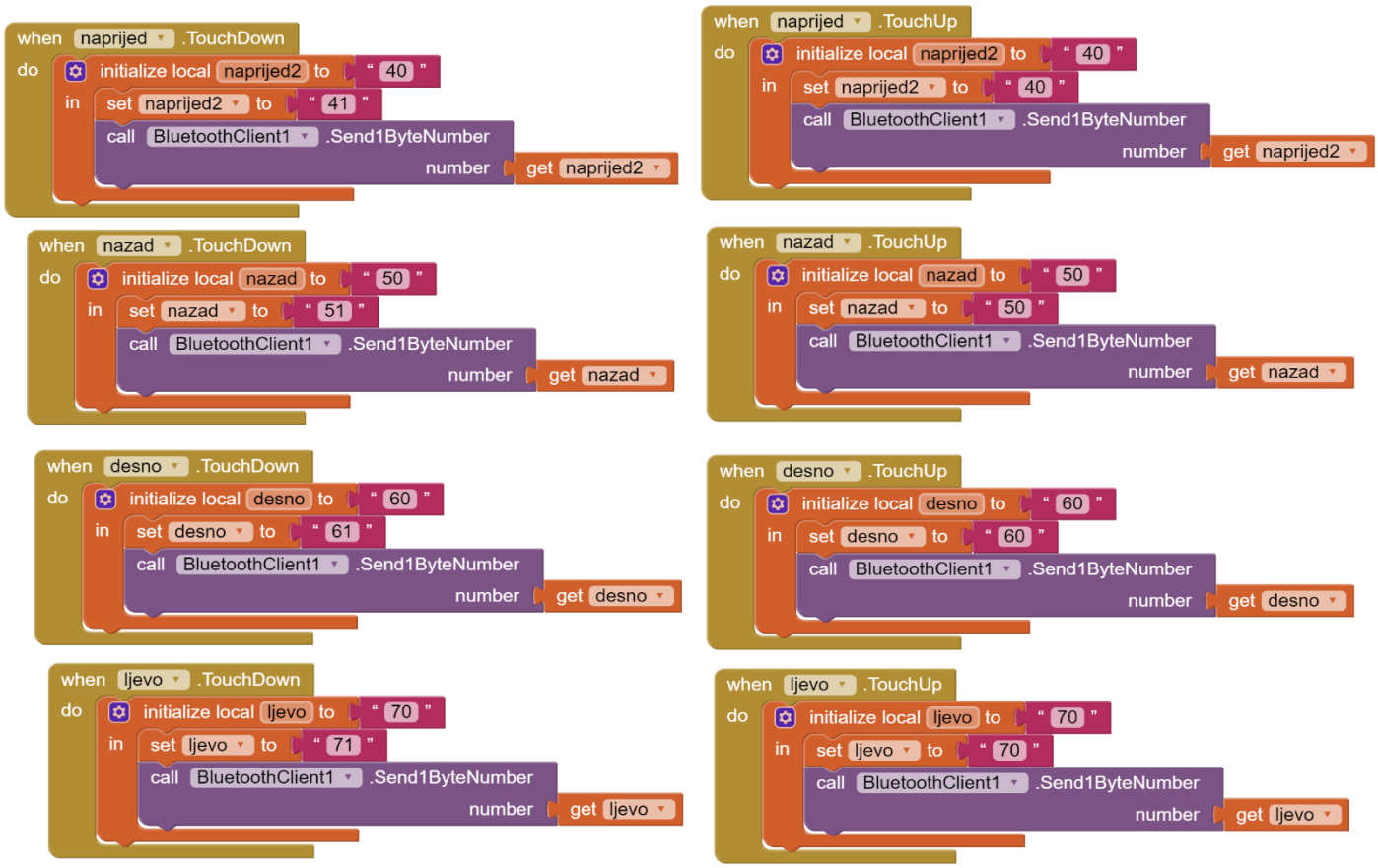
Slika 7.16. Prikaz blokovskog koda aplikacije.



Slika 7.17. Prikaz blokovskog koda aplikacije.



Slika 7.18. Prikaz blokovskog koda aplikacije.



Slika 7.19. Prikaz blokovskog koda aplikacije.