Automatizirani mobilni sustav za čišćenje podova s UZ senzorom

Šabić, Frane

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

MOBILNI SUSTAV ZA ČIŠĆENJE PODOVA S UZ SENZOROM

Završni rad

Frane Šabić

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:	Frane Šabić
Studij, smjer:	Stručni prijediplomski studij Elektrotehnika, smjer Automatika
Mat. br. pristupnika, god.	A 4563, 19.07.2019.
JMBAG:	0165080841
Mentor:	izv. prof. dr. sc. Tomislav Keser
Sumentor:	prof. dr. sc. Damir Blažević
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	izv. prof. dr. sc. Alfonzo Baumgartner
Član Povjerenstva 1:	izv. prof. dr. sc. Tomislav Keser
Član Povjerenstva 2:	doc. dr. sc. Tomislav Galba
Naslov završnog rada:	%naziv_rada%
Znanstvena grana završnog rada:	**Procesno računarstvo (zn. polje računarstvo)**
Zadatak završnog rada:	Projektirati, izraditi i testirati model sustava za automatizirano čišćenje prostora koji će se samostalno kretati u prostoru kojega čisti. Podsustav čišćenja izvesti na način da se koriste samo rotirajuće četke i/ili valjci. Prilikom kretanja prostorom treba imati mogućnost izmjere i pamćenja prostora kojim se kreće te mogućnost detekcije i izbjegavanja prepreka na putanji kretanja koristeći ulzrazvučni senzorski sustav. Upravljanje radom sustava omogućiti putem mobilne aplikacije ili web sučelja.

Datum ocjene pismenog dijela završnog rada od strane mentora:	18.06.2024.
Ocjena pismenog dijela završnog rada od strane mentora:	Izvrstan (5)
Datum obrane završnog rada:	16.7.2024
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:	16.07.2024.

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

IZJAVA O IZVORNOSTI RADA

Osijek, 16.07.2024.

Ime i prezime Pristupnika:	Frane Šabić
Studij:	Stručni prijediplomski studij Elektrotehnika, smjer Automatika
Mat. br. Pristupnika, godina upisa:	A 4563, 19.07.2019.
Turnitin podudaranje [%]:	9

Ovom izjavom izjavljujem da je rad pod nazivom**: Automatizirani mobilni sustav za čišćenje podova s UZ senzorom**

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

i sumentora prof. dr. sc. Damir Blažević

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

Od davnina ljudi teže ka tome da si na bilo koji način olakšaju život uporabom raznih pomagala i alata. Napredak u tehnologiji danas je vidljiv u svakom polju i život nam je lakši iz dana u dan jer su neki poslovi, koje bi inače fizički obavljali ljudi, zamijenjeni strojevima [1]. Takva se praksa proširila svim ljudskim djelatnostima, prvo u tvornicama gdje su tako stvorene pokretne trake, robotske ruke, kako bi se olakšao i ubrzao proces proizvodnje [1]. 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.

Model automatiziranog mobilnog sustava za čišćenje podova tehnologija je koja je nesumnjivo unaprijedila naše živote te su danas uistinu vrijedni pomoćnici u kućanskim poslovima [2]. Čišćenje je oduvijek bila jedna od najčešćih i nekim najtežih svakodnevnih zadaća kao i jedan često dugotrajan proces. Tako nije veliko iznenađenje da je potražnja za autonomnim robotima za čišćenje sve veća u modernom svijetu. Postoje modeli s različitim funkcijama, pa tako postoje oni koje je potrebno ručno pokrenuti, ali danas se već uglavnom radi o uređajima koji se mogu povezati s mobitelom i koji se tako prate putem određene mobilne aplikacije [3]. Na taj je način omogućeno njihovo beskontaktno pokretanje, praćenje i upravljanje na daljinu [4].

Automatizirani sustav za čišćenje mali je kućni robot i inteligentan uređaj koji ima vlastiti "mozak" s računalnom logikom kako bi mogao obaviti posao prema osmišljenom algoritmu [2]. Takvi mali automatizirani sustavi igraju važnu ulogu u svakom polju života jer se danas koriste i u industriji i u kućanstvima [4]. Činjenica je da se prosječan čovjek koristi sa 2 do 3 robota dnevno u svom svakodnevnom životu [4].

Sustav automatiziranog podnog čistača funkcionira putem nekoliko ključnih komponenti i senzora koji zajedno omogućuju učinkovito čišćenje podne površine. Glavne komponente uključuju:

- Procesorska jedinica (mikroupravljač): Ovo je temeljni dio sustava koji upravlja svim funkcijama čistača. Mikroupravljač prima podatke od senzora, obrađuje ih i upravlja aktuatorima prema programiranom algoritmu.

- Senzori: Senzori omogućuju čistaču navigaciju i prepoznavanje prepreka.
- Ultrazvučni senzori (UZ senzori): Koriste se za mjerenje udaljenosti od prepreka, emitiraju zvučne valove i mjere vrijeme potrebno za povratak odbijenog vala kako bi izračunali udaljenost do objekta.
- Infracrveni senzori (IR): prate rotaciju kotača i na taj način omogućuju izračunavanje prijeđene udaljenosti i praćenje pozicije.
- Senzori za prašinu: detektiraju količinu prašine i prljavštine na podu kako bi čistač prilagodio način rada.
- Aktuatori: Uključuju motore koji pokreću kotače i rotiraju četke.
- Pogonski motori: omogućuju kretanje čistača. Obično su to istosmjerni motori koji omogućuju precizno upravljanje brzinom i smjerom.
- Motor za četku: Rotira četke koje skupljaju prašinu i prljavštinu.
- Baterijski sustav: Napaja sve komponente čistača. Obično se koriste punjive litij-ionske baterije.
- Mobilna aplikacija: Omogućuje korisniku upravljanje čistačem, postavljanje rasporeda čišćenja i praćenje statusa u realnom vremenu. Aplikacija komunicira s čistačem putem bežične mreže točnije Wi-Fi-ja ili *Bluetooth*-a.
- Navigacijski sustav: omogućuje čistaču da se kreće po prostoriji i izbjegava prepreke. Najčešće se koriste SLAM (*Simultaneous Localization and Mapping*) tehnologija i predefinirani obrasci čišćenja.

Neki od primjera korištenja automatiziranih sustava za čišćenje su:

- Kućanstva – glavna namjena u kućanstvima im je automatsko čišćenje prašine, prljavštine i dlaka kućnih ljubimaca s podova. Funkcioniraju na način da koriste senzore za prepoznavanje prepreka i padova, algoritme za navigaciju kroz sobe i programibilne rasporede za čišćenje.
- Uredi - koriste se za redovito održavanje čistoće u uredskim prostorima, a funkcioniraju na način da ih se programira da čiste tijekom noći ili izvan radnog vremena kako ne bi ometali zaposlenike. Koriste navigacijske sustave za učinkovito čišćenje velikih prostora.

- Hoteli – korišteni za održavanje čistoće u zajedničkim prostorima i sobama za goste. Funkcioniraju na način da čiste hodnike i sobe prema rasporedu, omogućujući hotelskom osoblju da se fokusira na druge zadatke.
- Industrijski pogoni – korišteni za čišćenje velikih površina u proizvodnim halama i skladištima, rade na način da koriste senzore visoke preciznosti za navigaciju i izbjegavanje opasnih područja, a mogu biti prilagođeni za specifične vrste industrijskog otpada.
- Trgovački centri – koriste se kako bi održavali čistoću u trgovinama i zajedničkim prostorima i funkcioniraju na način da rade u noćnim satima ili izvan radnog vremena kako bi osigurali čistoću bez ometanja kupaca.
- Bolnice – korišteni kako bi osigurali visoke razine higijene u hodnicima i zajedničkim prostorima i opremljeni su dodatnim filtrima za prašinu i antibakterijskim sustavima za čišćenje. Rade prema strogim rasporedima kako bi minimizirali rizik od zaraza.
- Bazeni – korišteni za čišćenje bazenskih podnih površina, osim što su bitni kretanje i pozicioniranje, bitno je i zaštititi komponente od vode kako bi sustav uspješno radio.

Ovim radom htjelo se saznati više o dizajnu i zahtjevima jednog automatiziranog sustava za čišćenje prostora te projektirati i izraditi jedan model sustava za automatizirano čišćenje prostora koji se samostalno kretao u prostoru, pamtio prijeđen put i bio upravljan mobilnom aplikacijom.

1.1. Zadatak i struktura rada

Projektiranje, izrada i testiranje jednog modela sustava za automatizirano čišćenje prostora koji se potpuno samostalno kreće bio je glavni zadatak ovoga završnog rada. U tom su modelu rotirajuće četke korištene kao podsustavi čišćenja. Spomenuti model imao je i funkciju mjerenja i pamćenja prostora kojim se kreće, ali i funkciju detekcije i izbjegavanja prepreka koje mu se nalaze na putanji kretanja. Pri tome su korišteni ultrazvučni senzori. Putem mobilne aplikacije ili određenog internetskog sučelja bilo je moguće upravljati radom sustava.

Drugo poglavlje rada sadržano je teorijom koja je skrivena iza rješenja ovog zadatka, kao i pojašnjenjima funkcija svih postupaka koji će biti korišteni kako bi se riješio postavljeni zadatak. Također, u njemu je i detaljan opis sklopovskog rješenja s pripadajućim funkcionalnim blok

dijagramom. Osim sklopovskog, ovdje je dan i opisan i programski način rješavanja problema koji objašnjava sam tok algoritma koji je obavio rješenje zadanog problema.

U trećem poglavlju detaljnije su opisana sva predložena rješenja i njihova realizacija. Dani su kratki opisi korištenih alatnih aplikacija i programskih okruženja s detaljnom električnom shemom realizacije sklopovskog rješenja. Također, ovo poglavlje sadržava i detaljniji opis sklopovskog rješenja i njegova funkcioniranja na razini hardvera. Osim toga, ovdje je dan i detaljniji opis programskog rješenja s objašnjenim načinom funkcioniranja i detaljnim blok dijagramom toka programa.

Četvrto poglavlje sastavljeno je od prezentacije i objašnjenja samih testiranja i svih dobivenih rezultata. Ovdje je pobliže objašnjena sama metodologija kojom se pristupalo tijekom testiranja, pa su tako navedeni i pojašnjeni načini i postupci korišteni tijekom testiranja. Navedeno je što će biti testirano i na koji će način rezultati testiranja biti interpretirani. Rezultati testiranja prikazani su grafički ili dijagramom.

Peto poglavlje predstavlja zaključak cijeloga rada u kojemu je ukratko sumirano sve dotada rečeno, uz dodatak problema s kojima se susrelo tijekom realizacije i načina na koje su riješeni.

Šesto poglavlje jest popis literature i radova koji su korišteni u radu i na koje se kroz rad pozivalo.

2. SUSTAV ZA AUTOMATSKO ČIŠĆENJE PODOVA

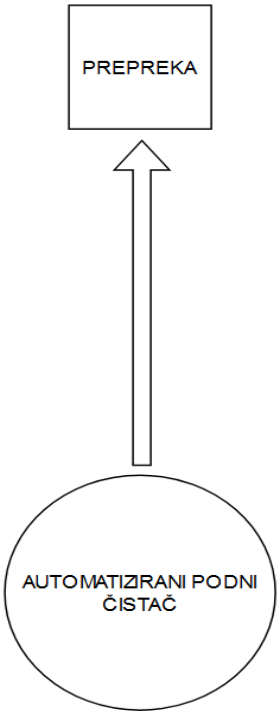
2.1.Teorijski osvrt na problem i rješenje rada

Zadani problem imao je pet temeljnih problema koje je potrebno riješiti. To su:

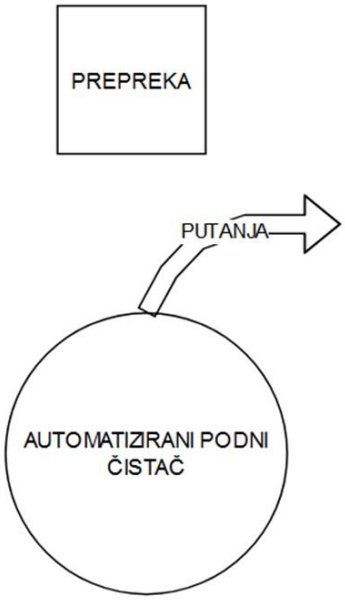
- prepoznavanje prepreka
- izbjegavanje prepreka
- praćenje prijeđenog puta
- čišćenje podne površine
- izrada mobilne aplikacije

2.1.1 Prepoznavanje prepreka i izbjegavanje istih

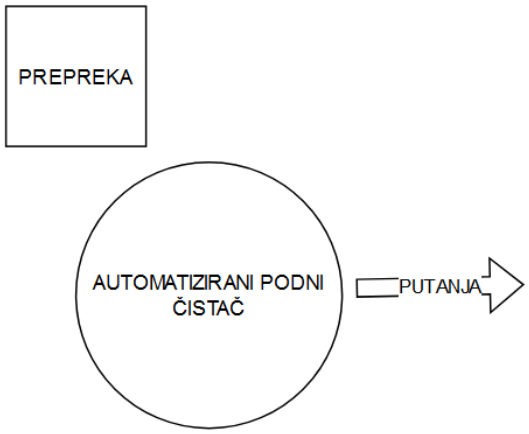
Prvi problem koji je nametnut je prepoznavanje prepreka i izbjegavanje istih zbog mogućeg kolizije s preprekom te zbog istog skretanje s putanje. Kako bi se kretao model automatiziranog sustava za čišćenje morao je imati mogućnost prepoznavanja predmeta i zapreka koje mu se nalaze na putu. Uz pomoć ultrazvučnog senzora, to se činilo na način da UZ senzor odašilje signal te prilikom primanja istog signala dobiva informaciju o udaljenosti prepreke. Kao što je to vidljivo na prvoj sljedećoj slici. Ako je udaljenost od prepreke manja od određene udaljenosti koja je bila izabrana prilikom testiranja rad motora će biti zaustavljen, slijedi zaustavljanje automatiziranog sustava za čišćenje. To je način prepoznavanja prepreke Nakon zaustavljanja slučajnim odabirom biran je smjer skretanja u lijevo ili desno te se skretanje izvodi uključivanjem lijevog ili desnog motora. To je vidljivo na drugoj slici. Treća slika prikazuje kretanje unaprijed nakon skretanja za određen kut u ovom slučaju 90 stupnjeva.



Slika 2.1. *Prikaz rješenja problema izbjegavanja prepreka tijekom kretanja prema prepreci.*



Slika 2.2. *Prikaz rješenja problema izbjegavanja prepreka prilikom stajanja i skretanja.*

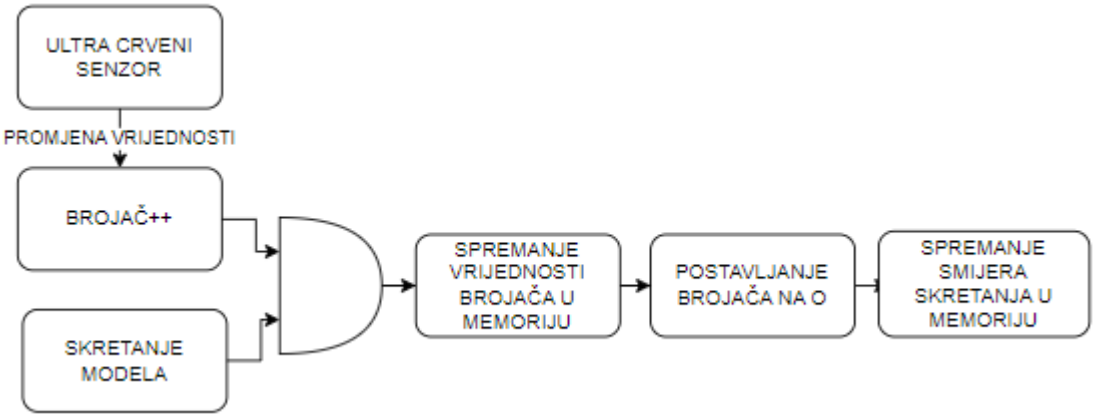


Slika 2.3. *Prikaz rješenja problema izbjegavanja prepreka nakon skretanja.*

2.1.2 Praćenje prijeđenog puta

Drugi problem koji je bilo potrebno riješiti jest i praćenje puta kojega je sustav prešao. Naime, to je problem jer je bilo potrebno indirektno izračunati vrijednosti pređene udaljenosti te kada i nakon koliko prijeđenog puta je model skrenuo lijevo ili desno. Infracrveni senzor omogućuje dobivanje informacije o tome je li predmet ispred njega s obzirom na njegovo kalibriranje pomoću potenciometra na način da šalje signal iz jedne LE diode, a drugom ju prima. Pomoću potenciometra je kalibriran da na iznimno maloj udaljenosti, od svega kojeg centimetra, prepoznaje ispred sebe predmet. Na osovinu motora koji okreće kotač stavljeno je pomagalo u obliku zvijezde sa što više krakova koji će se okretati sinkrono sa kotačem koje simulira okretomjer. LE diode infracrvenog senzora usmjeravane su na okretomjeru te je zbog toga infracrveni senzor davao pozitivan signal kada krak okretomjera bude ispred LE diode. Opseg kotača podijeljen sa brojem krakova okretomjera dao je duljinu koju je prešao podni čistač po svakom impulsu infracrvenog senzora. Sve to vidljivo je na slici ispod danog teksta. Prilikom

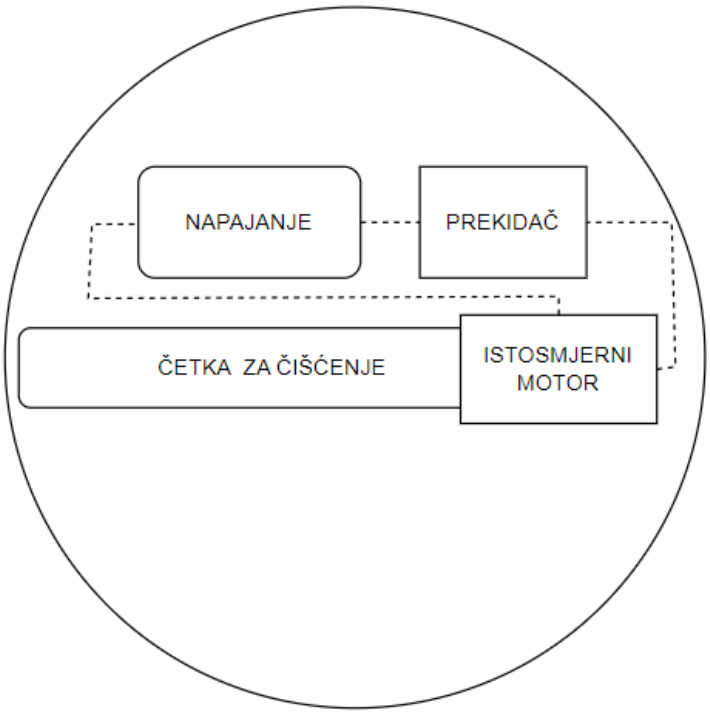
skretanja, u memoriju se spremala informacija o duljini do tada prijeđenog puta kao i smjer skretanja lijevo ili desno. Nakon spremanja u memoriju, brojač se resetirao kako bi se moglo započeti novo brojanje pređene duljine.



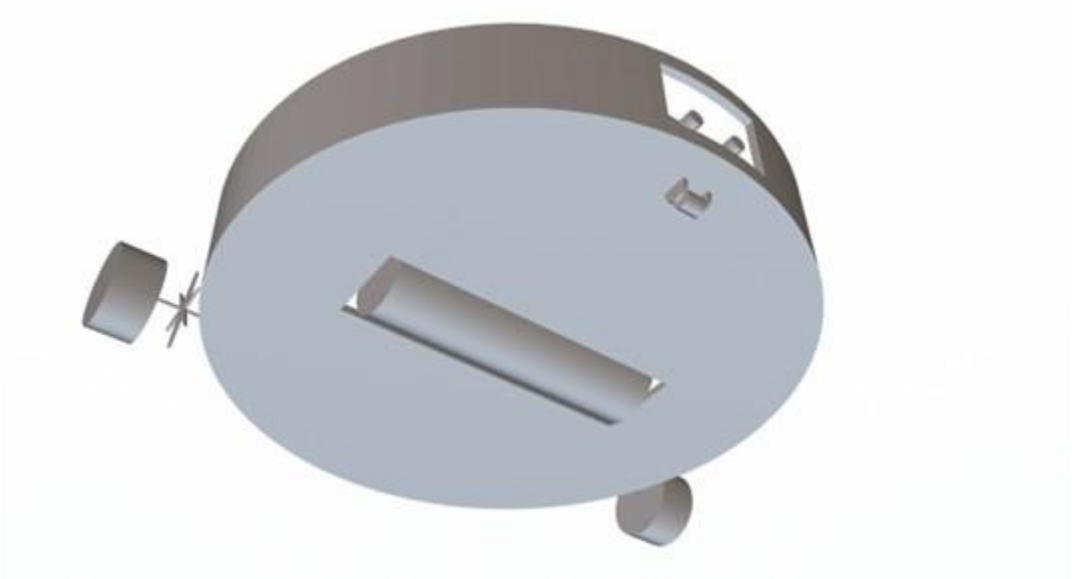
Slika 2.4 *Prikaz postupka praćenje prijeđenog puta.*

2.1.3 Čišćenje podne površine

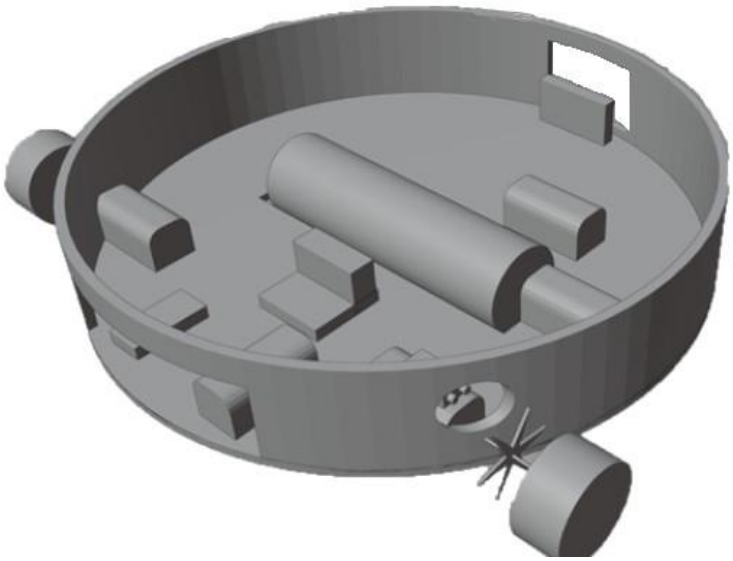
Čišćenje podne površine predstavljalo je problem zbog mnogobrojnih načina rješavanja. Odabran je način da to bude zaseban sustav, odnosno da metlica ili četka ima svoj vlastiti motor, svoje vlastito posebno napajanje te vlastiti prekidač za uključivanje i isključivanje sustava. Struktura je na dnu modela imala otvor kroz koju je četka dopirala do podne površine. Model četke je zamišljen da bude valjkastog oblika duljine do 20 cm kako bih pokrila što veću površinu. Na donjoj slici je prikazan strujni krug u kojemu je baterija predstavljena napajanjem. Istosmjerni motor služi kako bi dobivajući napon okretao spojenu metlicu pričvršćenu na njegov rotor, kako bi četka svojim okretanjem stvarala veću silu po podu te uklanjala nečistoće. Na slijedećim slikama 2.5. i 2.6. prikazan je 3D model kako je zamišljena pozicija četke te otvor s donje strane strukture modela čistača.



Slika 2.5. *Prikaz zasebnog sustava za čišćenje podne površine*.



Slika 2.6. *Prikaz 3D modela kojim je predstavljeno rješenje problema pozicioniranja četke kako bi ista dopirala do podne podloge.*



Slika 2.7. *Prikaz 3D modela kojim je predstavljeno rješenje problema pozicioniranja četke iz gornjeg ugla.*

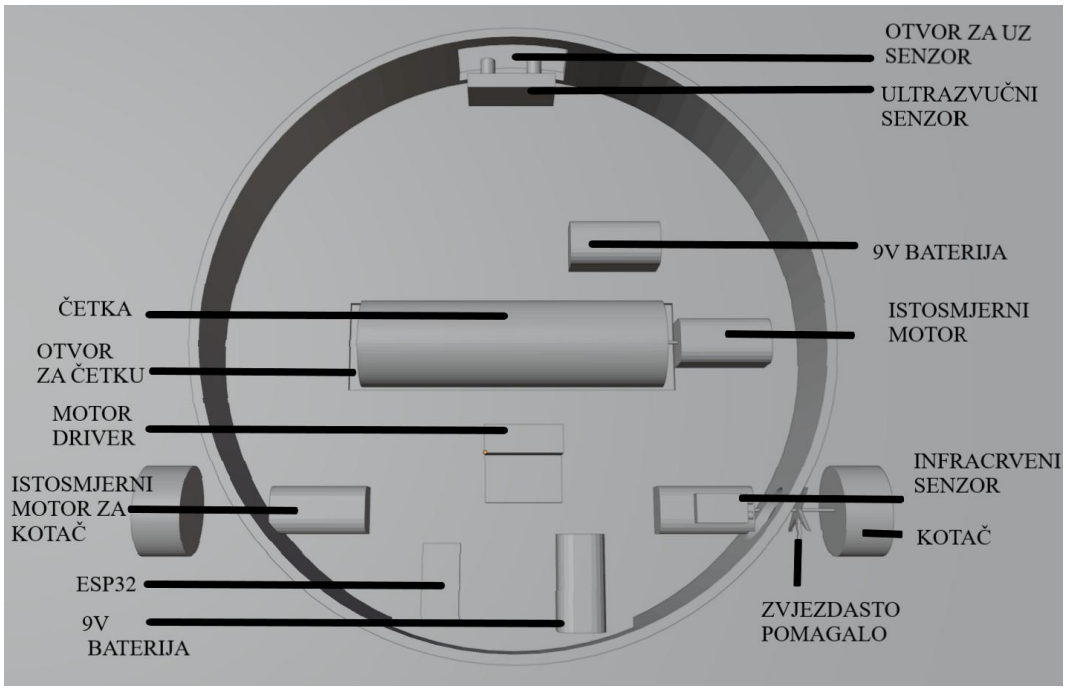
2.1.4 Izrada mobilne aplikacije

Treći problem predstavljala je izrada aplikacije. Bitno je bilo izabrati odgovarajući mikroupravljač koji ima mogućnost *bluetooht* spajanja. Izrada mobilne aplikacije naizgled je predstavljala problem visoke složenosti stoga je taj problem riješen uz pomoć *MIT app Inventor* platforme. Ta je platforma besplatna omogućuje izradu vlastitih mobilnih aplikacija blokovsko programiranje i integraciju već gotovih rješenja.

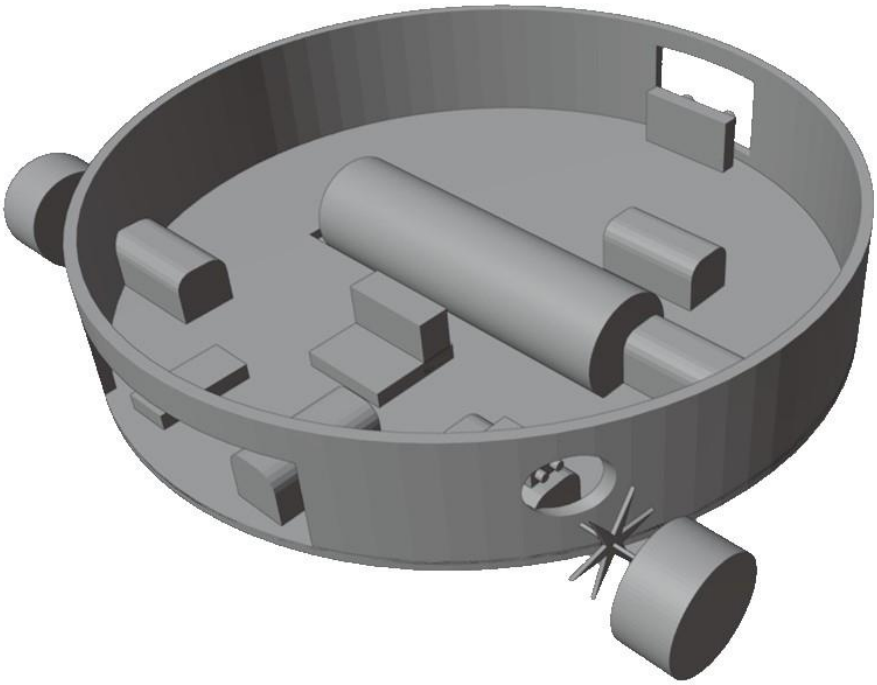
2.2. Prijedlog sklopovskog rješenja

Rad je bio zamišljen kao struktura koja je sadržavala elektroničke komponente automatiziranog sustava za čišćenje podova izradi od plastičnih dijelova temeljenih na 3D modelima pomoću 3D

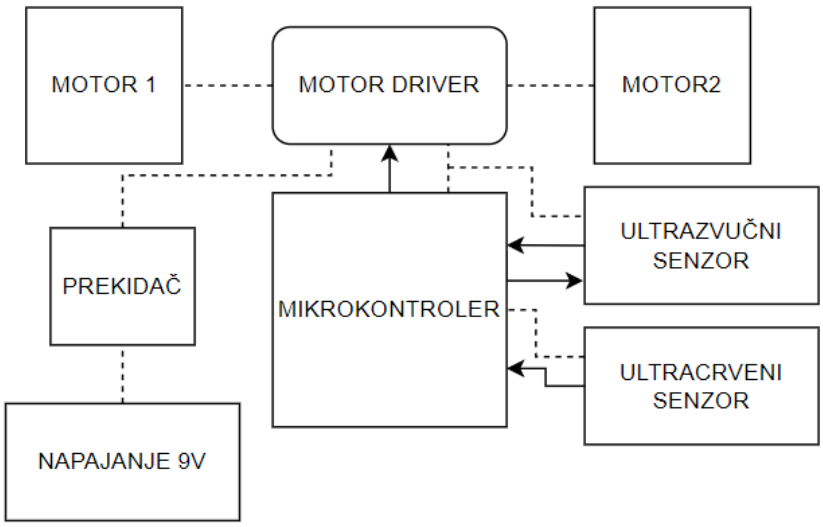
pisača. Model je zamišljen u obliku valjka s promjerom baze nešto preko 20 cm i visinom oko 5 cm. Takav oblik se nametnuo kao najbolja solucija zbog lakšeg prolaza kroz prostor, odnosno zbog minimalne mogućnosti da zapne za neki dio namještaja. Na tako ispisanom modelu s prednje strane je bio otvor za pričvršćeni ultrazvučni senzor koji je služio za izbjegavanje prepreka. Model je također imao dva otvora sa stražnje strane kako bi se omogućilo spajanje i mijenjanje koda na mikroupravljač, kao i punjenje 9V baterije koja je napajala cijeli sustav. Na gornjem dijelu su bile biti dvije sklopke za uključivanje i isključivanje sustava, odnosno za početak i kraj čišćenja. One su se nalazile između baterije i ostatka sustava. Nakon baterije, pomoću sklopke za uključivanje i isključivanje, bio je motor driver koji je dijelio napon istosmjernim motorima, mikroupravljaču i ultrazvučnom senzoru.



Slika 2.8. 3D p*rikaz konstrukcije zamišljenog modela sustava za čišćenje podne površine s njegovim komponentama*.

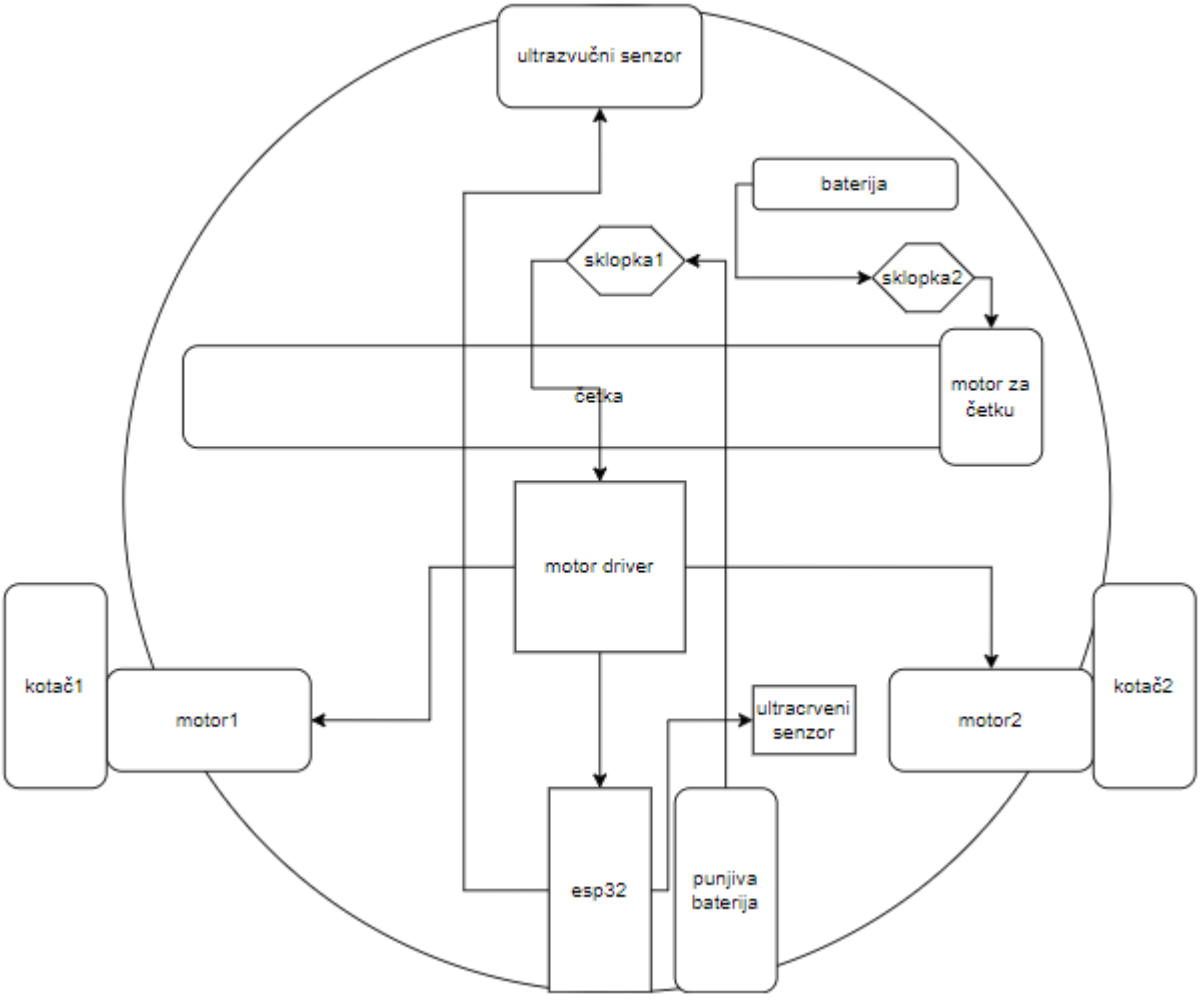


Slika 2.9. 3D p*rikaz konstrukcije zamišljenog modela sustava za čišćenje podne površine* iz *drugog kuta kako bi se vidio sustav za praćenje prijeđenog puta*.



Slika 2.10. *Blokovski prikaz sklopovskog rješenja*.

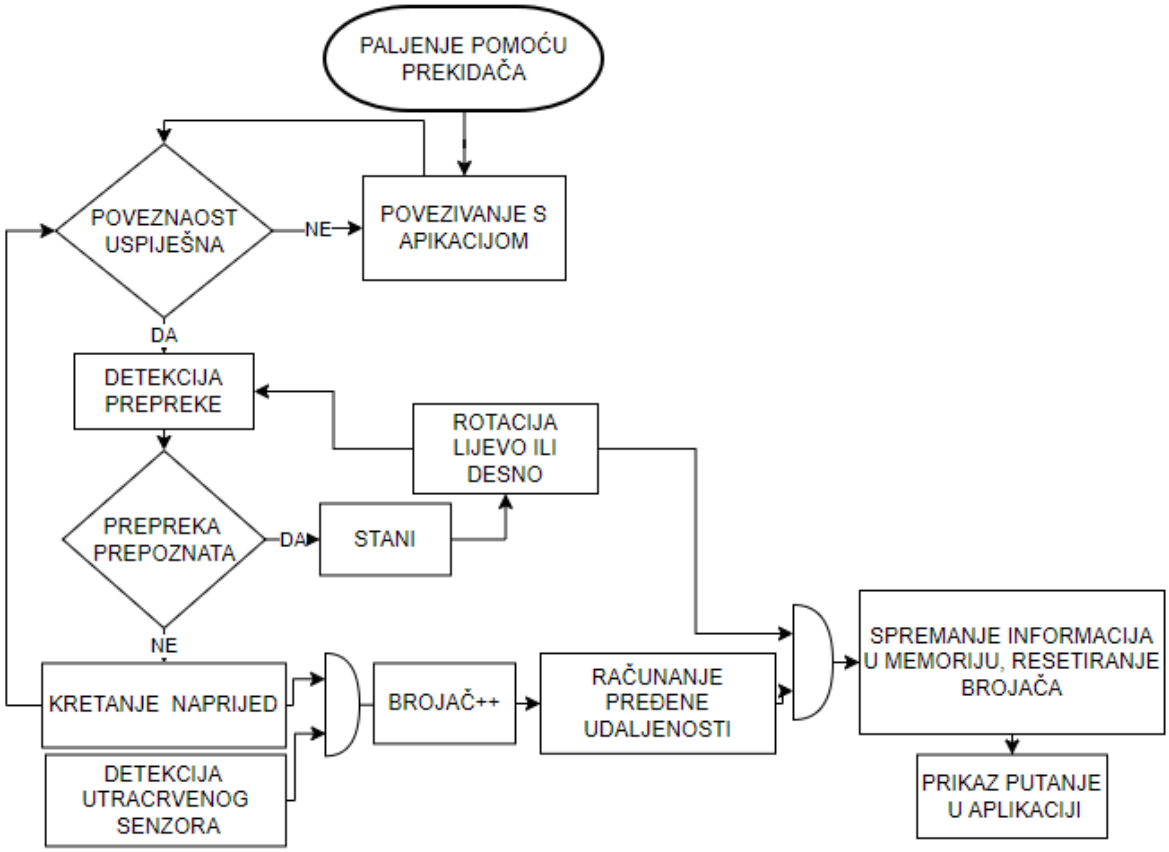
Za kretanje su bila korištena tri kotača, od kojih je jedan pomoćni dok su ostali pogonski. Pogonski kotači bili su pokretani istosmjernim motorima napajanim putem motor drivera, a njihova osovina spojena s kotačima praćena infracrvenim senzorom pomoću kojeg se izračunavala prijeđena udaljenost. Infracrveni senzor napajan je i upravljan mikroupravljačem. Između kotača bila je smještena četka kojoj je zadatak čišćenje podne površine. Četka je pokretana zasebnim istosmjernim motorom koji je napajan zasebnom baterijom, a između se nalazilo tipkalo za uključivanje i isključivanje tog kruga.



Slika 2.11. *Prikaz strukturne raspodjele komponenata unutar čistača.*

2.3. Prijedlog programskog rješenja

U programskom dijelu nakon inicijalizacije konstanti, pomoćnih varijabli, ulaza i izlaza dolazi se do *loop* (petlja koja se vrti) dijela. U tom dijelu petlje, program radi na sljedeći način. Udaljenost od prepreke se iščitavala ultrazvučnim senzorom i uspoređuje s predviđenom zadanom udaljenosti. Ako je prepreka dalje od te iste udaljenosti, pozivana je funkcija za kretanje naprijed tijekom čega se broje signali koje daje infracrveni senzor uz određenu obustavu vremena (*delay*). Iz tog broja se preračunavala prijeđena udaljenost pomoću informacije o opsegu kotača i ta informacija je spremljena u memoriju. Kada je prepreka na kraćoj udaljenosti od zadane udaljenosti, pozivala se funkcija zaustavljanja te nakon određenog čekanja, slučajnim odabirom se biralo hoće li se zvati funkcija za okret u desno ili u lijevo. Obje te funkcije traju jednaku količinu vremena, što je vrijeme potrebno modelu za okret od 90 stupnjeva. Tijekom okretanja, brojač se resetirao na nulu i u memoriju se spremio smjer rotacije modela.



Slika 2.12. *Pojednostavljeni prikaz dijagrama toka mikroupravljača.*

Programska aplikacija prije svega započinje uspostavom povezivanja s *bluetooth* modulom iz našeg čistača. Nakon toga pritiskom sklopke u aplikaciji omogućen je rad modela. Prilikom svakog skretanja na pozadini mobilne aplikacije prikazane su primljene vrijednosti koje predstavljaju prijeđenu dužinu čistaća te smjer u kojemu se kretao.

3. REALIZACIJA MOBILNOG SUSTAVA ZA ČIŠĆENJE PODOVA

U sljedećim poglavljima pobliže će biti objašnjeni i nabrojani programski alati i komponente korištene u izrađivanju zadanog automatiziranog sustava za čišćenje. Osim toga, detaljnije će biti opisana i realizacija sklopovskog rješenja, ali i kako je realizirano samo programsko rješenje.

3.1. Korišteni alati i programska okruženja

Prilikom izrade automatiziranog sustava za čišćenje kao programski alati korišteni su Arduino IDE i *MIT app Inventor*. Arduino integrirano razvojno okruženje je korišteno s uređivačem teksta za pisanje koda, područje za poruke, tekstualnu konzolu, alatnu traku s gumbima za uobičajene funkcije i niz izbornika. Zahvaljujući jednostavnom i pristupačnom korisničkom iskustvu, Arduino je korišten u tisućama različitih projekata i aplikacija. Softver Arduino je jednostavan za korištenje za početnike, ali dovoljno fleksibilan za napredne korisnike. Radi na Macu, Windowsu i Linuxu. Hardver i softver Arduino dizajnirani su za umjetnike, dizajnere, hobiste, hakere, početnike i sve zainteresirane za stvaranje interaktivnih objekata ili okruženja. Arduino može biti komuniciran s tipkama, LE diodama, motorima, zvučnicima, GPS jedinicama, kamerama, internetom, pa čak i pametnim telefonom ili televizorom. Jedna od glavnih prednosti Arduino IDE je jednostavnost korištenja. Arduino IDE ima jednostavan i intuitivan *editor* koji podržava označavanje sintakse, automatsko dovršavanje i formatiranje koda. Također može biti korišten ugrađeni serijski monitor i crtač za otklanjanje pogrešaka i vizualizaciju podataka. Arduino IDE olakšava učitavanje koda na uređaj samo jednim klikom, bez brige o upravljačkim programima, *kompajlerima* ili konfiguracijskim datotekama. Međutim, Arduino IDE također ima neka ograničenja koja mogu utjecati na njegovu upotrebljivost za naprednije korisnike. Arduino IDE ne podržava napredne značajke kao što su restrukturiranje postojećeg djela koda, alati za otklanjanje pogrešaka, kontrola verzija ili testiranje jedinice. Također ima ograničeni sustav upravljanja knjižnicom koji može uzrokovati probleme s kompatibilnošću. Osim toga, nisu dopuštene prilagodba ili optimizacija koda za specifične zahtjeve hardvera ili performansi.

MIT *App Inventor* je intuitivno, vizualno okruženje za programiranje kojim je omogućena izrada potpuno funkcionalnih aplikacija za Android telefone, iPhone i Android/iOS tablete. Oni novi u MIT *App Inventoru* mogu postaviti i pokrenuti jednostavnu prvu aplikaciju za krače od 30 minuta. I štoviše, ovaj alat temeljen na blokovima olakšava stvaranje složenih, visokoučinkovitih aplikacija u znatno kraćem vremenu od tradicionalnih programskih okruženja. *MITApp Inventor* je besplatna usluga temeljena na oblaku koja vam omogućuje izradu vlastitih mobilnih aplikacija pomoću programskog jezika temeljenog na blokovima. Platforma *MIT App Inventor* nastoji demokratizirati razvoj softvera osnaživanjem svih ljudi, posebno mladih, da prijeđu s potrošnje tehnologije na stvaranje tehnologije. Neke od prednosti MIT-a:

- Sve je rađeno putem jednostavnog odabira. To znači da može biti odabran određeni dio koda i u to ubaciti naš kod. Dakle, nema tipkanja.
- Lako testiranje novostvorene aplikacije.
- Korisnicima su pružene neke osnovne lekcije koje pomažu u izradi aplikacija i koje pomažu u ispravnom razumijevanju načina na koji platforma MIT-a radi.
- Korisno za početnike.
- Snaga izvornih aplikacija s jednostavnim korisničkim sučeljem

Glavni nedostaci mogu se vidjeti u smanjenoj fleksibilnosti u domeni rješavanja problema, nepostojanja korisničkog sučelja i složenog okruženja početnika [6].

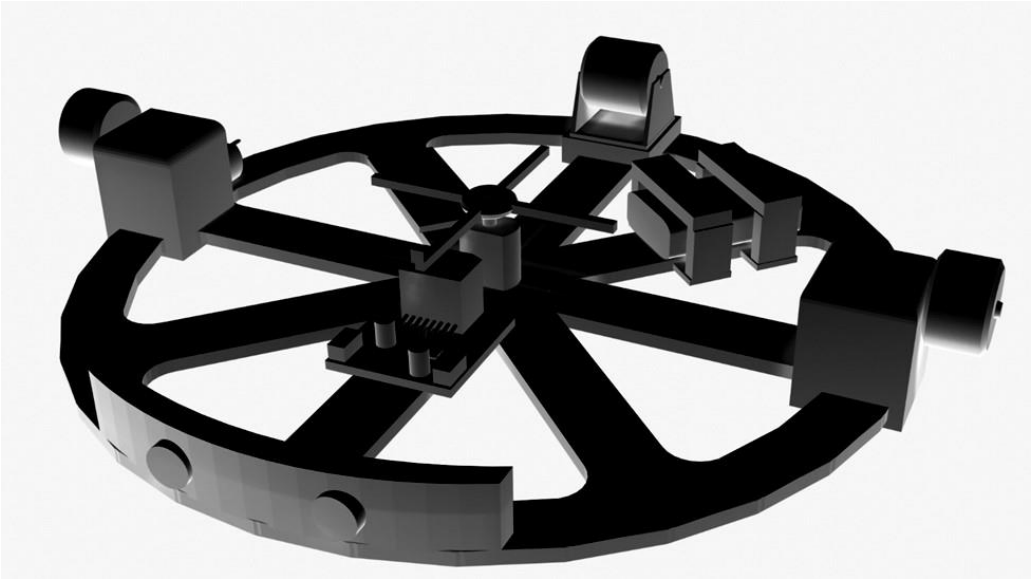
Od ostalih sklopovskih komponenata korišteni su:

- ESP32 wroom generički modul
- motor driver L298n
- 9V punjiva baterija
- infracrveni senzor TCRT5000
- ultrazvučni senzor HC-SR04
- istosmjerni motor.

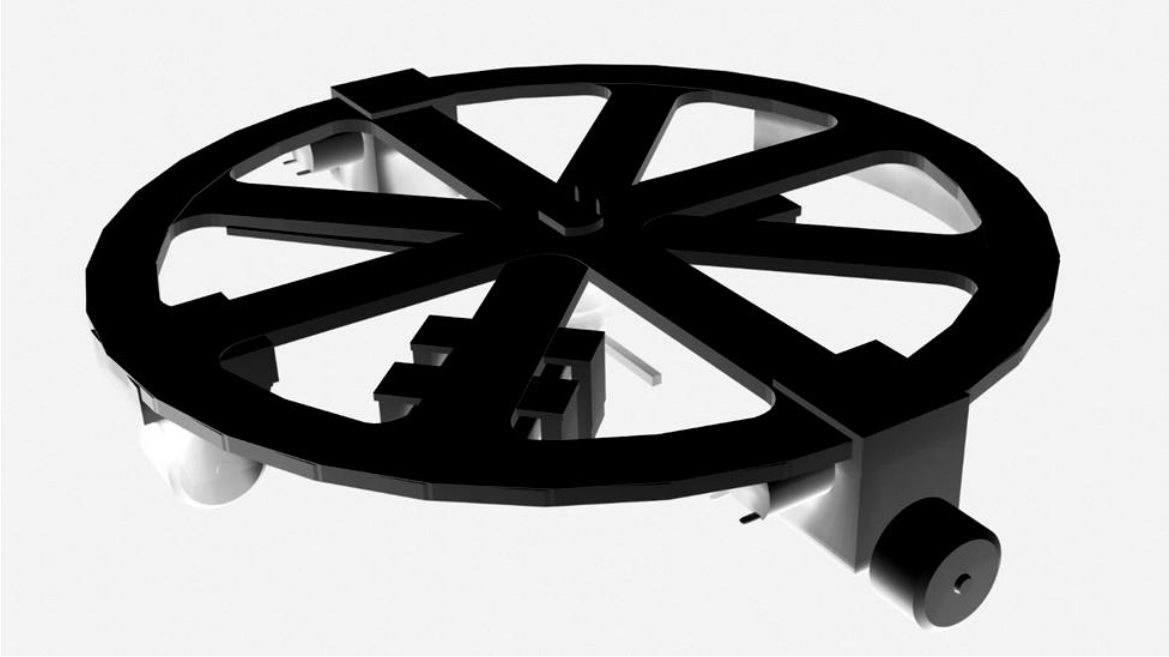
3.2. Realizacija sklopovskog rješenja

Automatizirani čistač sačinjen je od elektroničkih komponenti kojima se upravlja mikroupravljačem ESP32-Wroom koji je detaljnije objašnjen u poglavlju 7.1.2. Prema prikazanoj shemi na mikroupravljač bio je spojen ultrazvučni senzor HC-SR04 (pojašnjen u poglavlju 7.1.3.). Pomoću podataka dobivenih sa senzora, podatak o udaljenosti od prepreke je u svakom trenutku dostupan automatiziranom čistaču, pa je tako omogućeno nesmetano kretanje unutar prostorije bez kolizije s preprekama. Jedna punjiva baterija i tri 1,5V baterije za motor metlice korišteni su kao izvor napajanja čistača (7.1.4.). Korišteni su i tri istosmjerna elektromotora, jedan koji pokreče četkicu, a druga dva za kretanje. Smjerom vrtnje dva istosmjerna elektromotora mikroupravljač je upravljao pomoću l298N Motor Drivera (7.1.1.). Na kosturu istosmjernih motora za kretanje bili su pričvršćeni IR senzori koji nam omogućavaju praćenje prijeđenog puta. Napajanje samog mikroupravljača i ultrazvučnog senzora se obavljalo putem izlaza od 5 volti l298N Motor Drivera. Mikrupravljač je programiran u Arduino IDE razvojnom okruženju, a električna shema dizajnirana u alatu EasyEDA prikazana je na slici 3.2.

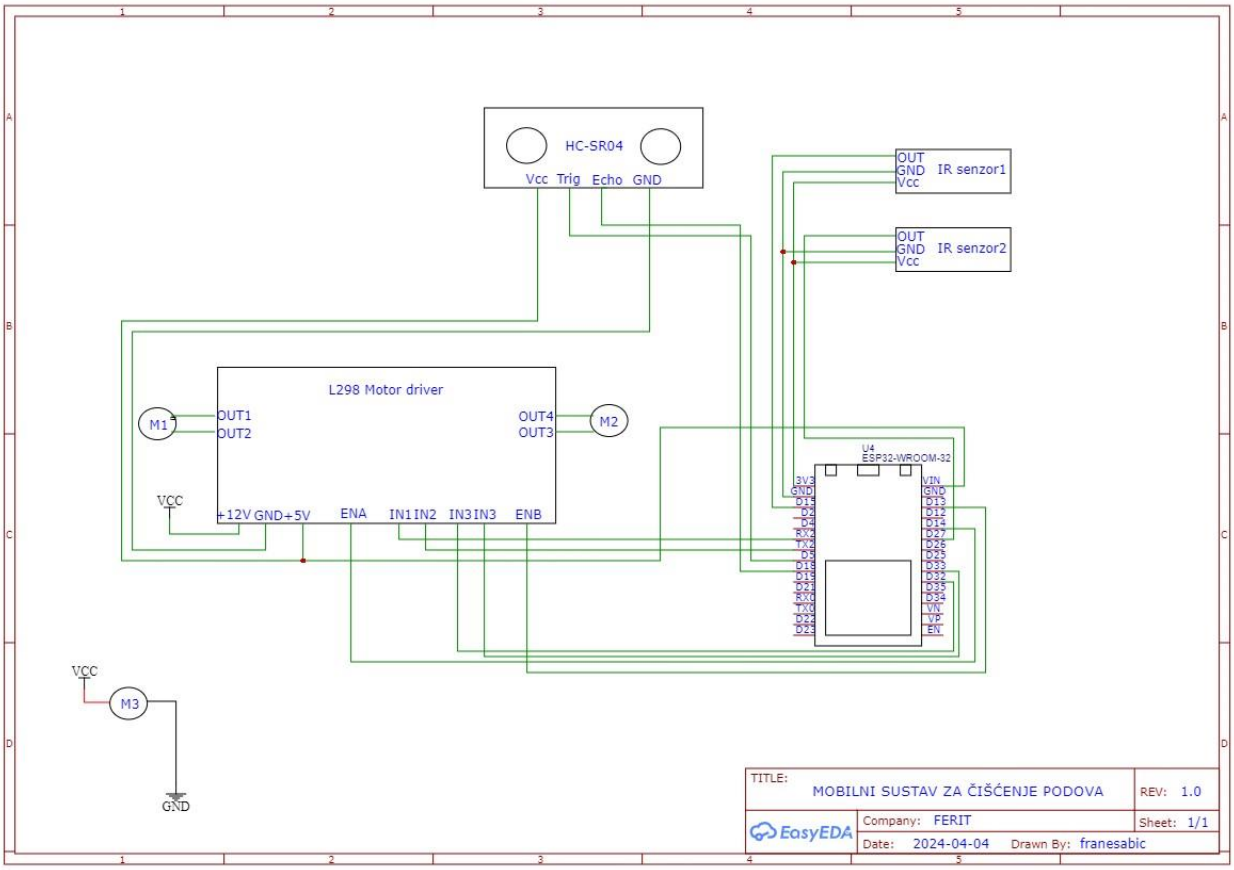
Došlo je do velikih promjena u odnosu na konstrukcijsku shemu iz poglavlja 2.2 iz razloga što nije bilo moguće raspodijeliti komponente na 3D ispisanu bazu automatiziranog čistača kao što je to prikazano na shemi iz poglavlja 2.2.



Slika 3.1. *Prikaz cjelovitog 3D modela podnog čistača s donje strane u Blenderu.*



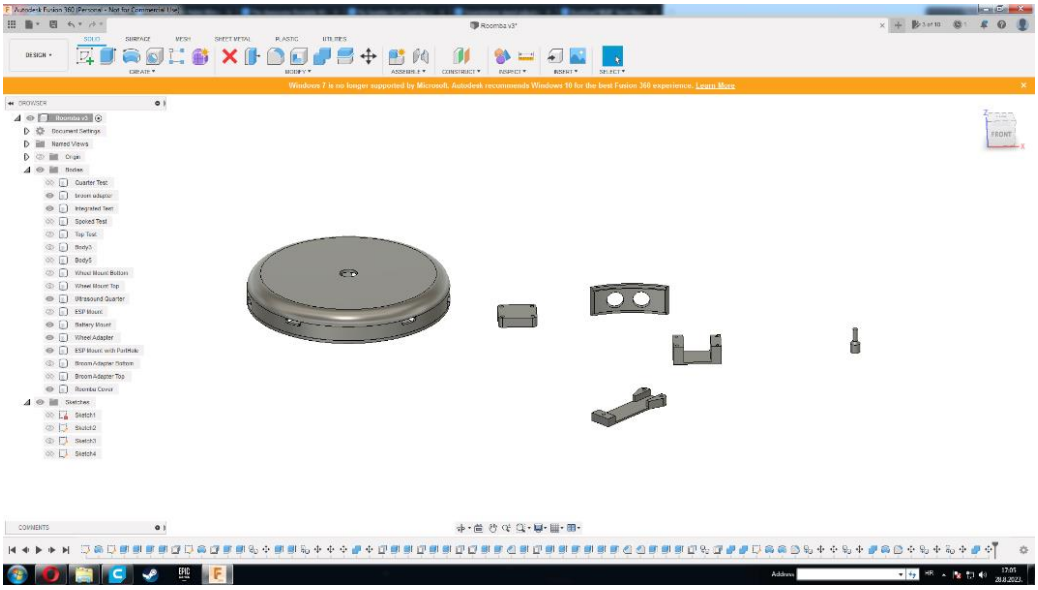
Slika 3.2. *Prikaz cjelovitog 3D modela podnog čistača s gornje strane u Blender-u.*



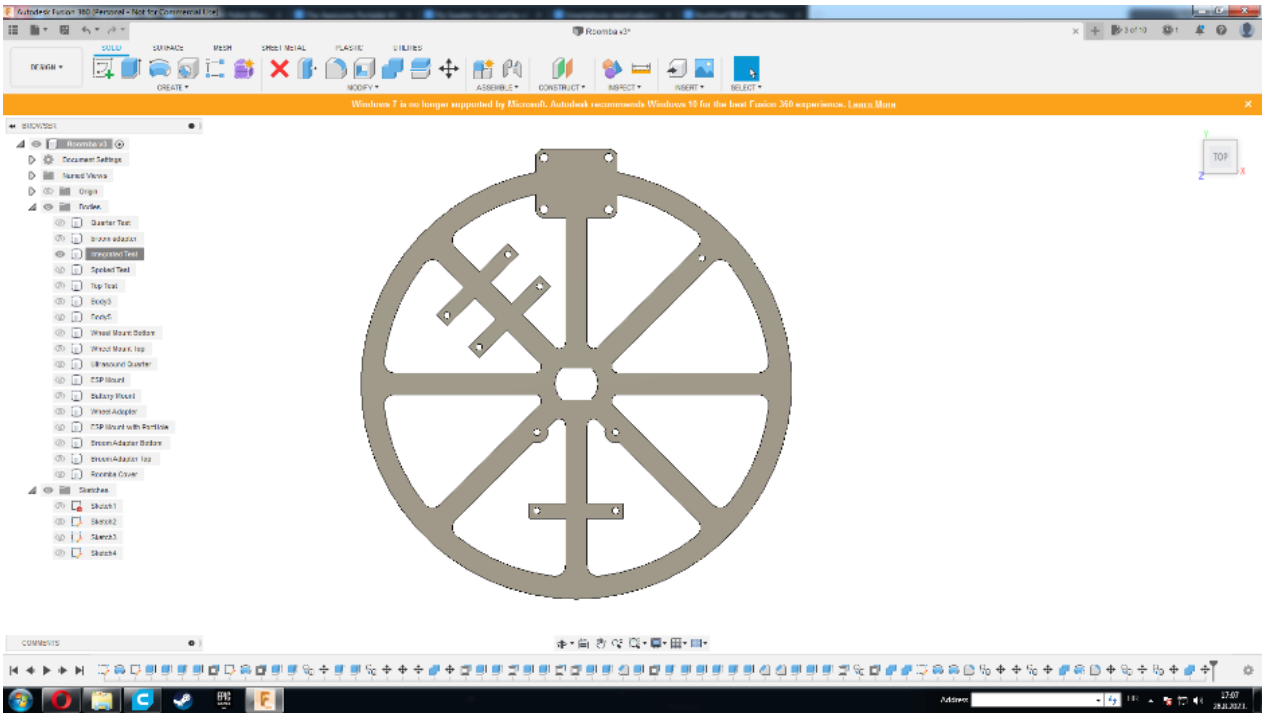
Slika 0.3. *Električna shema dizajnirana u alatu EasyEDA.*

Koraci pri izradi automatiziranog čistača:

1. Prvi korak bio je odabiranje najboljeg materijala za izradu čistača. Odabran je 3D ispisni plastični materijal iz razloga što daje mogućnost odabira oblika i dimenzija kućišta, dovoljno je čvrsta i lagana te je ujedno i najjeftinija opcija. Računalni dizajn kućišta izrađen je u *Fusion 360* programu. Kućište je izrađeno od dva glavna dijela: baze i poklopca, a manji 3D ispisani dijelovi zalijepljeni su specijalnim ljepilom za plastiku s bazom. Za razliku od drugog dijela, gdje je predviđeno korištenje četke u obliku valjka, od tog plana se odustalo zbog prevelike veličine same četke. Zbog toga ne bi postojala mogućnost dizajniranja kućišta iz jednog komada zbog nemogućnosti ispisivanja dimenzija tih veličina. Zbog gore navedene odluke o korištenju metlice, a ne četke ispisivana je b3D osovina koja spaja serijski motor (7.1.5.) i metlicu. Za istosmjerni motor koji pokreće metlicu ostavljeno je mjesta u samoj sredini kućišta. Odabran je oblik koji nije puni disk kako bi se uštedilo na materijalima i težini kao i zbog lakšeg pričvršćivanja ostalih komponenti i prostora za same vodiče. Također, razlika u odnosu na drugi dio je ta što je treći kotač postavljen iza radi ravnoteže i lakše raspodjele pozicije komponenata.



Slika 3.4. *Prikaz svih elemenata (osim baze kućišta) u Fusion 360.*

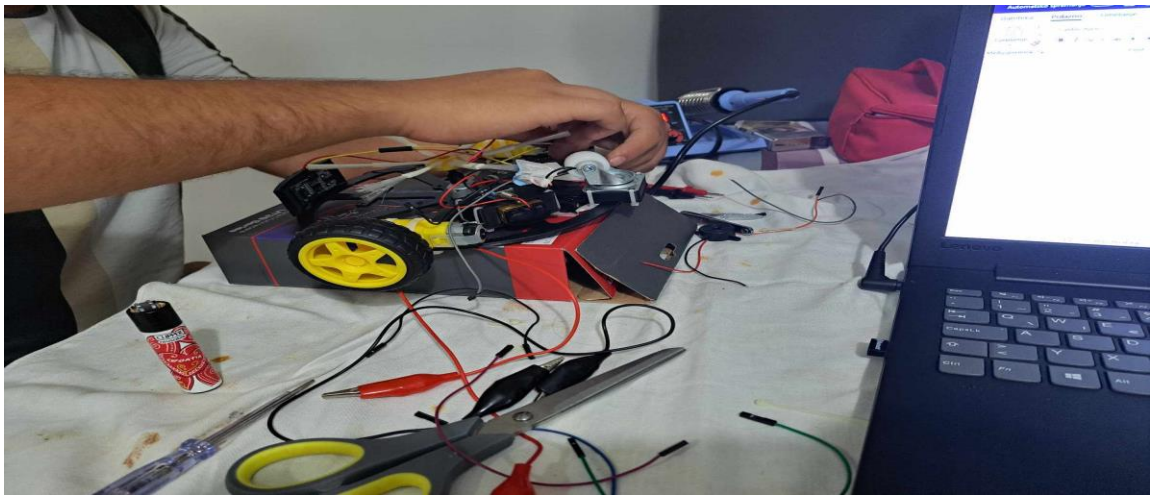


Slika 3.5. *Prikaz baze kućišta u Fusion 360.*

2. Drugi korak bio je postavljanje komponenti na unaprijed određena mjesta na bazi, komponente su vezicama učvršćene za bazu. Za svaki istosmjerni motor bile su potrebne dvije vezice dok je pomoćni učvršćen sa četiri vezice. Držači baterija koji su 3D ispisani osigurani su s 5 vezica, kao i držač mikroupravljača koji je osiguran s tri vezice. Poslije ovog najjednostavnijeg koraka pri izradi čistača pojavljuje se problem pri spajanju prekidača. Problem je polazio od toga što nije ispisano dovoljno unaprijed određenih mjesta za njih prilikom ispisivanja. Odluka rješenja tog problema bila je zalijepiti prekidače na držače baterija sa specijalnim ljepilom za plastiku. Ultrazvučni senzor nije bilo potrebno učvrstit vezicama jer je za njega ispisan držač koji je isto tako dodatno učvršćen specijalnim ljepilom za plastiku. Zbog odluke da se neće koristiti četka u obliku valjka već metlica s šest krakova, istosmjerni motor postavljen je vertikalno kako bi se rotirao paralelno s podlogom koju čisti. Isti je pričvršćen specijalnim ljepilom za plastiku te na 3D ispisanu osovinu. Istom je metodom osovina pričvršćena na metlicu. Kalibracija visine četke je bila iznimno bitna kako čistač ne bi zapinjao metlicom na površinu što bi u konačnici ometalo kretanje. Sklapanje elemenata i lijepljenje kućišta prikazano je na slici ispod teksta.

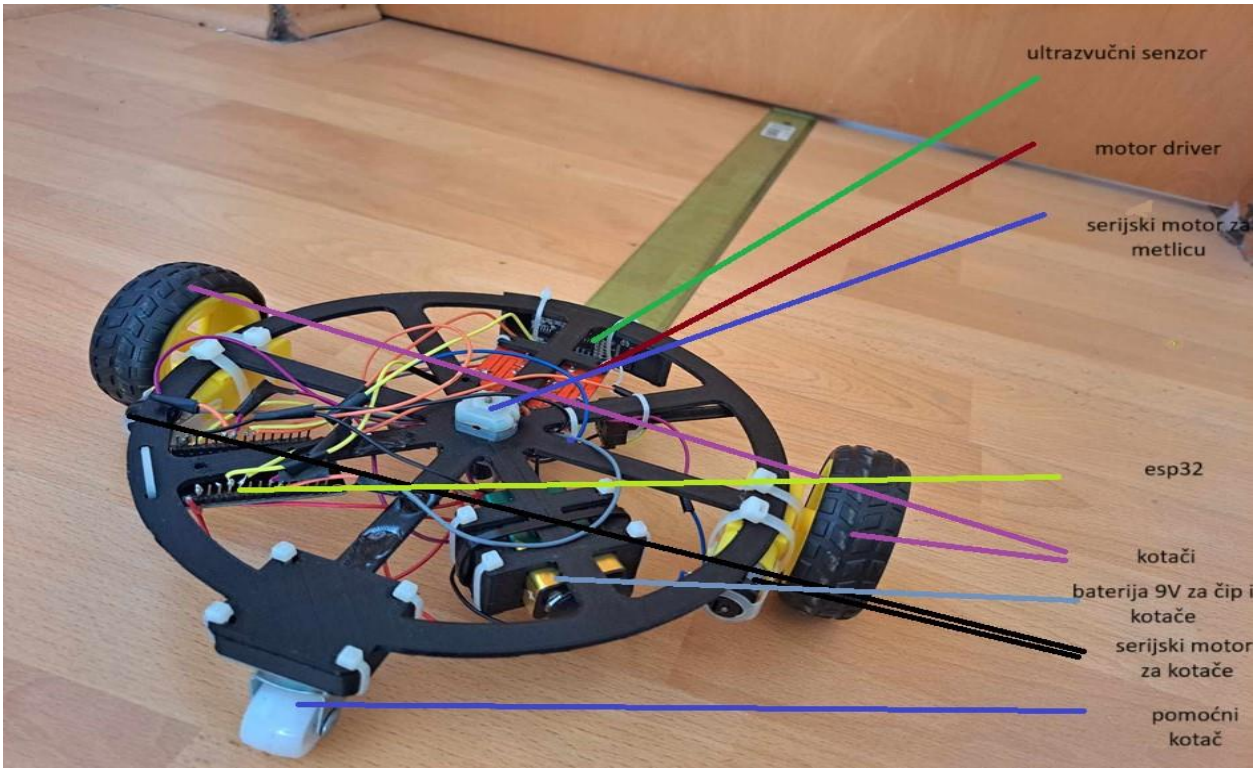
Slika 3.6. *Sklapanje komponenata i lijepljenje kućišta.*

3. Nakon drugog djela bilo je potrebno spojiti komponente. Najprije je spojena punjiva baterija s prekidačem te je isti sklop spojen s L298n motor driverom. Iz motor drivera izlaze vodiči prema unaprijed pričvršćenim motorima. Tu je bilo bitno obratiti pozornost na koji izlaz je spojen koji *pin* motora kako bi se motor vrtio u željenom smjeru te kako kasnije tijekom procesa ne bi bilo potrebno raditi dodatne promjene u kodu. Kako je mikroupravljaču potreban napon od 5V spojen je na izlaz od 5V i uzemljenje L298n motor drivera. Zadnji korak u ovom djelu je bio spojiti nožice odnosno *pin*-ove ESP-A na ultrazvučni senzor i sami motor driver s motorima. Povezivanje *pin*-ova prikazano je ispod na slici 3.5.



Slika 3.7. *Povezivanje pin-ova.*

4. Prije nego što je sve zalemljeno, odrađena je provjera samog čistaća te je bilo vidljivo njegovo prebrzo kretanje. Rješenje tog problema planirano je ispuniti potenciometrom koji bi regulirao motore PWN regulacijom no, shvaćeno je da je bespotreban potenciometar te da je istu regulaciju moguće obavljati samo softverom odnosno mijenjajući vrijednosti na PWN-u.



Slika 3.8. *Testiranje opisano u 4. koraku*.

5. Nakon postavljanja komponenti na red je došlo lemljenje i spajanje komponenti vodičima. Paralelno je obavljeno testiranje komunikacije komponenti s multimetrom. Ovo je bilo otežano zbog toga što se moralo voditi računa o tome da vodiči ne smiju biti prepreka vrtnji četkice. Nakon uspješnog lemljenja, prikazanoga na slici 3.6. postavljena je izolacijska traka kako ne bi došlo do slučajnog doticanja između izloženih zalemljenih površina vodiča. Zadnji korak bio je postavljanje ispisanog poklopca radi zaštite komponenti, smanjene mogućnosti vanjskih utjecaja te u konačnici i izgleda samog sustava.

Slika 3.9. *Lemljenje pin-ova.*

6. Nakon što je napravljen hardver za kretanje došlo je na red spajanje infracrvenih senzora namijenjenih za indirektno praćenje prijeđenog puta. Isti senzori su potom zalemljeni s predviđenim naponom od 3,3V, za njih, kao i na uzemljenje. Također, uslijedilo je spajanje odnosno lemljenje signalnih *pinova* senzora koje je potrebno spojiti na unaprijed određene pinove

ESP-a. Poslije uspješnog povezivanja IR senzora potrebno je pronaći prikladno mjesto za njih kako bi mogla biti dobivena informacija o okretanju kotača. Pričvršćeni su vezicama za kućište motora kotača te su usmjereni na kotač zbog strukture kotača koji su zvjezdastog oblika.

7. Nakon što je IR senzor spojen potrebno je isti kalibrirati. Kalibracija je vršena tako što je prije svega usmjerena dioda koja šalje infracrvenu zraku da cilja u prostor između krakova zvjezdastog kotača, a dioda koja prima infracrvenu zraku usmjerena je pod određenim kutom prema diodi koja šalje zrake, ali na način da ne bude previše savijena prema diodi koja šalje kako se ne bi dobio konstanto isti signal. Nakon namještanja dolazi se do kalibriranja potenciometra koji kalibrira osjetljivost infracrvenih senzora odnosno udaljenost na kojoj je predmet očekivan.

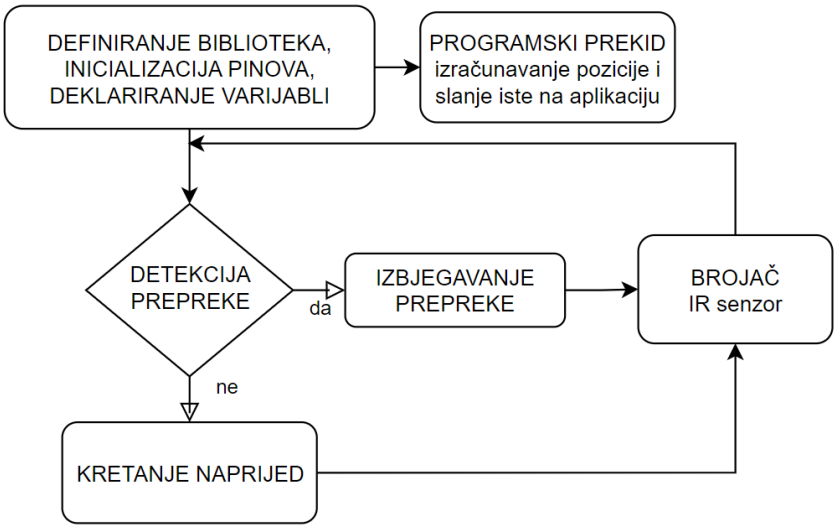
3.3. Realizacija programskog rješenja

U ovom dijelu objašnjen je kod ArduinoIDE-a u poglavlju 3.3.1. koji predstavlja programskog čistača bez aplikacijskog dijela, dok je aplikacijski softver rađen u MitApp Inventoru i objašnjen u 3.3.2.

3.3.1. Realizacija programskog rješenja samog podnog čistača

Na početku programa (koda) rješenja završnoga rada objašnjenom u poglavlju 7.2 Na početku programa (koda) rješenja završnoga rada objašnjenom u poglavlju 7.2 potrebne biblioteke su definirane, pinovi su inicijalizirani i potrebne varijable su definirane. Ovaj dio je detaljnije opisan u tekstu poslije prvog i osnovnog dijagrama, poglavlju 3.3.1.1. Nakon ovog nužnog dijela dolazi se do *interupta* (tj. Programskog prekida objašnjemom u dijelu 3.3.1.2.) koji je pozivan svakih 0.1 sekundu, te je tako izračunata prosječna brzina zadnjih 0.5 sekundi kao i pozicija podnog čistača u kojoj se trenutno nalazi. Nakon toga te su informacije poslane putem *bluetooth*-a u aplikaciju. Više o tome u nastavku teksta ispod u poglavlju 3.3.1.3. Nakon što je definirana funkcija za kretanje i *setup* (u dijelu 3.3.1.1.), definiran je *loop* dio programa odnosno dio programa koji se konstantno izvršava. Na početku *loop*-a učitana je vrijednost s potenciometra te je ta vrijednost kalibrirana i poslana na *pinove* motor drivera, a služi za limitiranje brzine vrtnje istosmjernih motora u funkciji kretanja prema naprijed. Zatim je *TRIGER pin* ultrazvučnog senzora postavljen

na *HIGH* kako bi poslao ultrazvučni val koji se odbija od prepreke i vraća u *receiver* koji postavlja *ECHO pin* na *HIGH*, zatim se pomoću ugrađene funkcije *pulseIn* dobiva vrijednost koja se onda pohranjuje u varijablu *duration, a* koja se množi s polovicom konstante brzine zvuka kako bi konačno bila dobivena vrijednost koja predstavlja udaljenost u centimetrima. Ona je pohranjena u varijablu *distance*. Onda dolazi do same logike kretanja s obzirom na detektiranu udaljenost. Ako predmet bude detektiran pomoću ultrazvučnog senzora na manjoj udaljenosti od postavljene, postupak zaobilaženja prepreke (koji je objašnjen zadnjim u dijelu 3.3.1.5.) je bio pokrenut, a ako nije detektiran nastavljalo se s funkcijom *GoForward* koja je objašnjena u dijelu 3.3.1.4. Osim funkcija za kretanje, tu je također i funkcija za IR senzore koja prati okretaje motora i objašnjena je u poglavlju 3.3.1.2.



Slika 3.10. *Detaljan blok dijagram toka programa.*

3.3.1.1. Definiranje biblioteka, incijalizacija *pinova*

Na početku programa (koda) završnoga rada objašnjenu u poglavlju 7.2., uključuju se biblioteke za *Bluetooth, za interrupt* (programski prekid) *(freertos)* i za trajnu memoriju *NVS* kako bi iste mogle biti pravovremeno pozvane. Poslije biblioteka deklarirani su pinovi za ultrazvučni senzor, infracrvene senzore, pogonske motore i PWN kontrolu pogonskih motora, za pogonske motore. Također, poslije toga deklarirane su varijable *duration* i *distance* koje su korištene za

pohranjivanje podataka potrebnih ultrazvučnom senzoru. Varijable *BrojPromjenaStanja1* i *BrojPromjenaStanja2* postavljene su na 0 koje predstavljaju broj promjena na IR senzorima. Kao i *predhodnjoStanje1, predhodnjoStanje2* postavljene na „LOW“ radi usporedbe promjene stanja na IR senzorima. Pored ovih varijabli deklarirane su 4 varijable za millis funkciju koja je korištena za promjenu brzine kretanja bez da pri tome vremenski ometamo ultrazvučni i ultracrveni senzor u glavnom *loopu*. Nakon što su deklarirani segmenti za ultrazvučni senzor, infracrveni senzor i millis funkciju, dolazi se do deklariranja varijable koja *handle*-a timer za programski prekid. Slijedi deklariranje varijabli koje služe za izračunavanje pozicije podnog čistaća u programskom prekidu. X i Y predstavljaju trenutnu poziciju podnog čistača, dx dy pomak čistača od posljednjeg programskog prekida, c prosječnu brzinu u zadnjih 5 programskih prekida, te dosta pomoćnih varijabli koje su objašnjena u dijelu 3.3.1.2. U *setup*-u su sva četiri *pina* za upravljanje motorima postavljena na *output. Serial begin* odnosno brzinu komunikacije je postavljena na 115200 *bauda*. U *setup*-u su također *pin*-ovi postavljeni za ultrazvučni senzor, t*riger* na *output*, a *Echo* na *input*. Onda je u kodu kalibrirana očitana vrijednost te je ista poslana na motor driver kao informacija za upravljanje istim. Osim toga i imenovanja podnog čistača na *bluetooth* komunikaciji inicijalizirana je i NVS memorija te je obrisana stara memorija odnosno memorija od posljednjeg korištenja podnog čistača. Stvara se *timer* koji će svakih 0.1 sekundu pozvati programski prekid te je isti aktiviran.

3.3.1.2. Praćenje okretaja kotača pomoću IR senzora

Slijedeći dijagram koji predstavlja ažuriranje senzora počinje sa provjerom je li IR senzor na *high* ili *low*. Nakon toga uspoređena je ista vrijednost s prošlom, ako su različite povećava brojač za jedan te postavlja prethodnu vrijednost na trenutnu vrijednost. Taj postupak brojio je na svakom kotaču promjenu stanja koja je bila potrebna za izračunavanje pozicije podnog čistača koja se obavljala u dijelu koji se izvodio tijekom programskog prekida.

Slika 3.11. *Detaljan dijagram funkcije rada IR senzora.*

3.3.1.3. Programski prekid

U programskom prekidu započinje se s razlikom broja promjena stanja te množenjem s 2.1 kako bi bila dobivena razlika u cm. Pomoću te razlike I znanja da je centar kotača udaljen 13 cm računa se smjer skretanja podnog čistača od zadnjeg programskog prekida. Bitna je stavka da je smjer kut između tangente imaginarne kružnice koju podni čistač prati gledajući da je jedan kotač u sredini kružnice, a drugi prati istu kružnicu i normale koja prolazi kroz kotač koji stacionira u centru kružnice. Nakon što je dobiven zadnji smjer u radijanima isti je ažuriran tako što je zbrojen s dosadašnjim zbrojem smjerova odnosno ažuriran je smjer. Pomoću istog smjera I pređene udaljenosti podnog čistača u cm, koja je izračunata s prosjekom promjena puta konstantom koja je kalibrirana s obzirom na radijus kotača I broja promjena stanja kako bi bila dobivena prijeđena veličina u cm, slijedi izračunavanje pomaka na x,y osi s obzirom na poslijednju poziciju. Bitno je napomenuti, ako je pomoćna varijabla za kretanje unazad aktivna na 1 onda se kretanje x,y računa

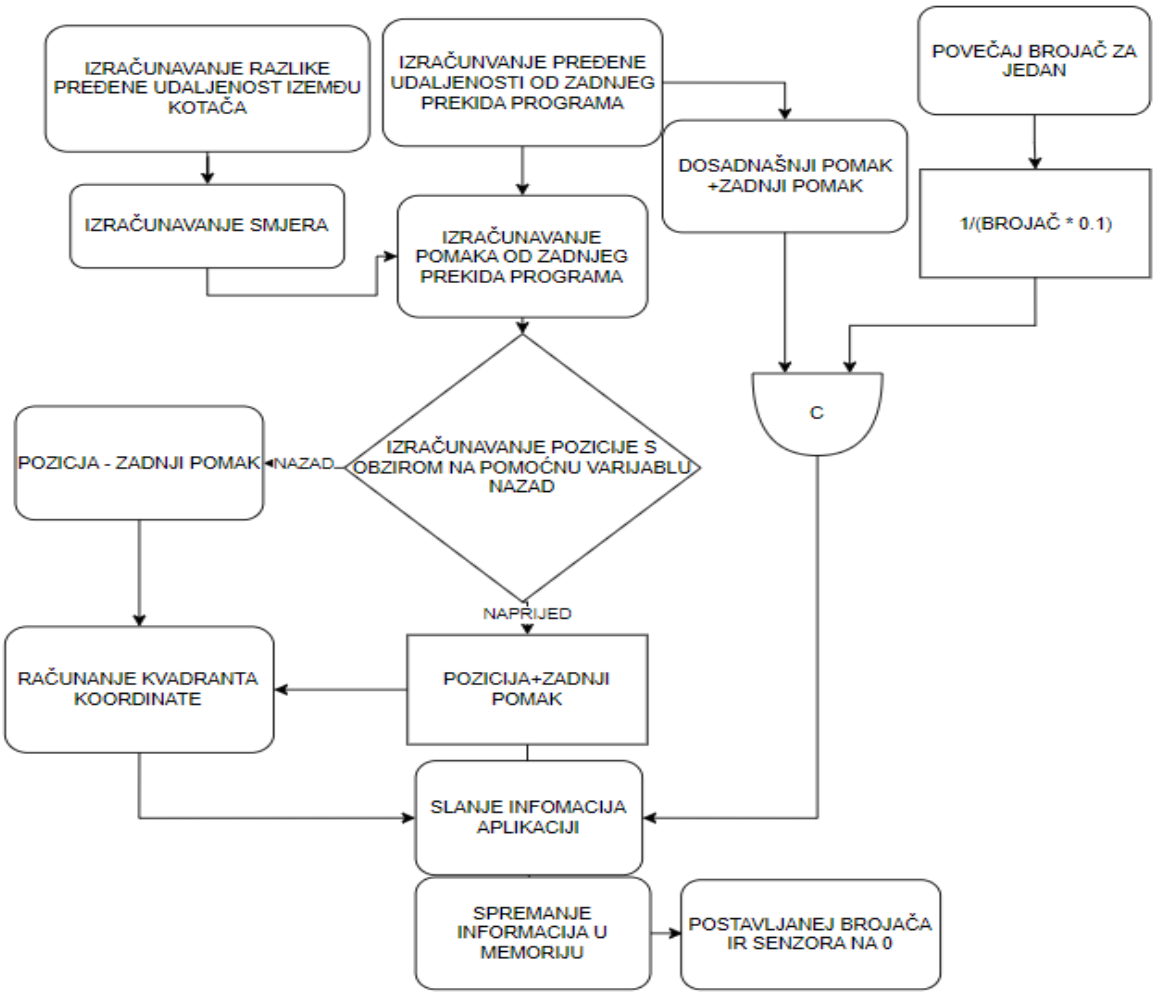
s negativnim predznakom kako bi se zbrajajući x,y s posljednjim *x,y* dobila trenutna lokacija podnog čistača. U dijelu 3.3.1.5. je navedeno kada se mijenja pomoćna varijabla. S obzirom na xy izračunavamo kvadar na koordinatnoj osi u kojoj se nalazi, a u specifičnoj situaciji kada je x ili y jednako 0 stavljamo u prvi odnosno treći kvadrant. Ova informacija je bila od presudne važnosti za pravljenje aplikacije te je u tom dijelu rada bolje pojašnjeno.

Računanje brzine izvedeno je na način da je stavljen brojač koji se poveća svaki *interupt*, a također se povećava pomoćna varijabla koja zbraja svaku prosječnu predenu duljinu s dosad prijeđenim putem u cm dijelimo s 0.1s *brojačem *ineterupta*. Tako je dobivena prosječna brzina od starta do sada.

X,y,z (koji predstavljaju kvadrant u kojem se podni čistač nalazi) i C (koji predstavlja brzinu). Stavljen je *string* u obliku *x,y,z,c* te je poslan *blutetooth*-om aplikaciji.

Kako bi bilo moguće spremiti x,y u memoriju također u obliku *string-a* napravljen je brojač pomoću kojeg se x i y informacije spremaju svaki put na drugu adresu.

Nakon svega resetirana je vrijednost broja promjena stanja na oba senzora na nulu.

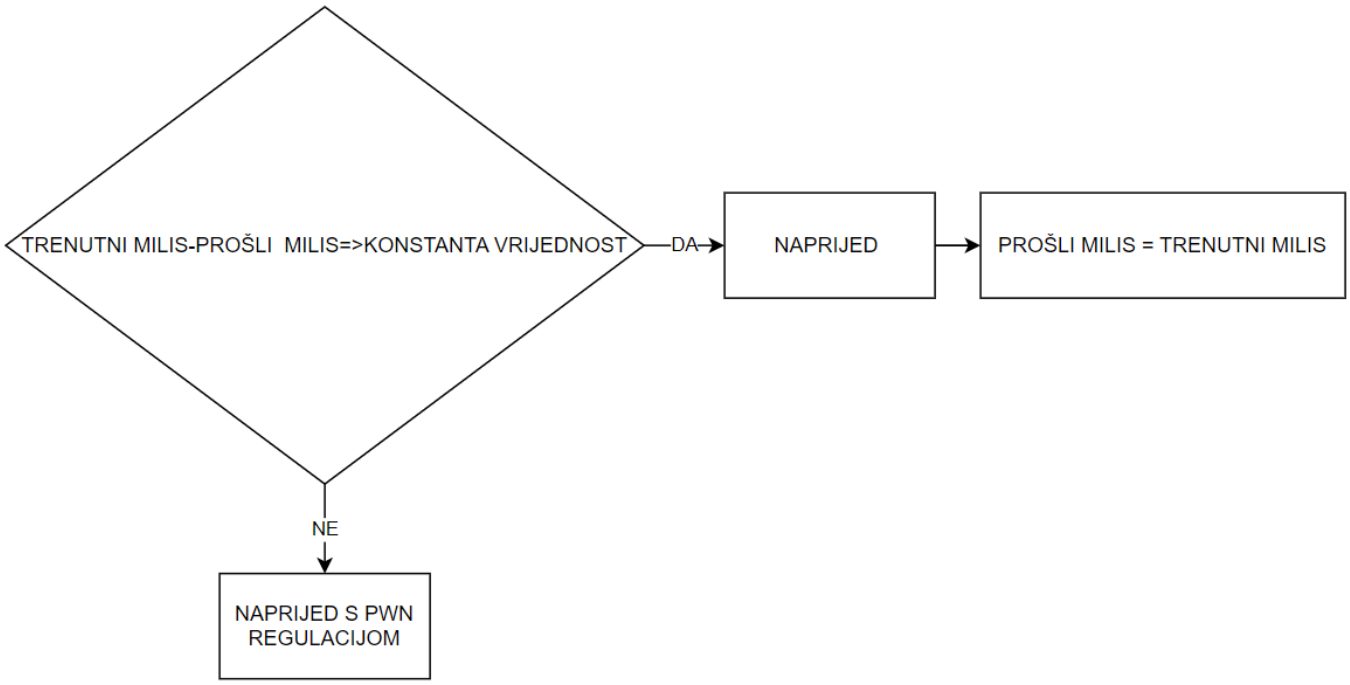


Slika 3.12. *Detaljan dijagram uzračunavanja pozicije brzine i slanje na aplikaciju u programskom prekidu.*

3.3.1.4. Kretanje unaprijed

U dijelu *GoForward* odnosno u dijelu koji je pojašnjen na slijedećoj slici vidljiva je varijabla u koju je spremljena funkcija millis. U „*if*“ petlji oduzet je od iste varijable prošli millis koji je prije postavljen na 0. Ta je razlika onda uspoređena s intervalom koji predstavlja vrijeme pod kojim će se pozivati funkcija *GoForward* te će se podni čistač kretati naprijed. *GoFoward* funkcija u sebi ima digitalne *pinove* za motore postavljene na valjane vrijednosti kako bi se kretao naprijed i ena i enb analogne *pinove* postavljene na prije kalibrirane vrijednosti kako bi bilo moguće regulirati

brzinu pomoću PWN. Kada prođe taj interval pozvana je funkcija Go*foward2* koja je radila isto kako i Go*foward* samo što se može gledati kao da nema PWN regulaciju odnosno da je išla maksimalnom mogućom brzinom. Kretanje naprijed napravljeno je na način cikličkog mijenjanja *GoForward* i *GoForward2* kako bi bilo izbjegnuto kružno kretanje podnog čistača u slučaju da jedan kotač ne dobije dovoljno snage kako bi pokrenuo jedan motor.

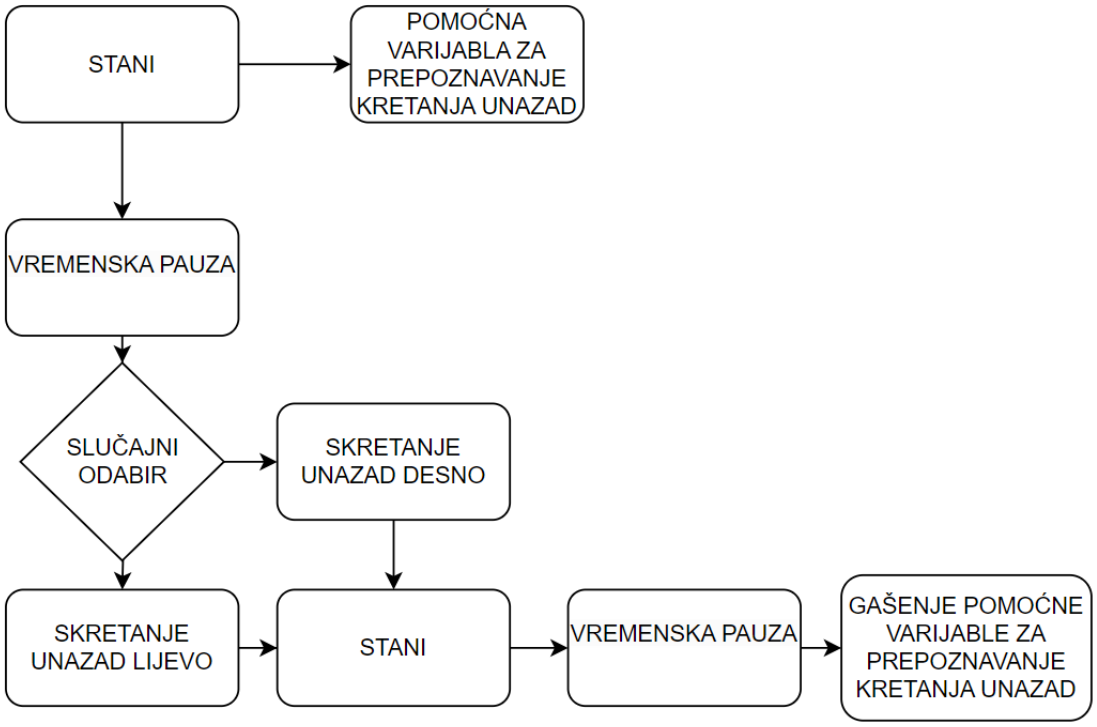


Slika 3.13. *Detaljan dijagram funkcije millis koja omogućava kretanje naprijed pomoću dvije funkcije.*

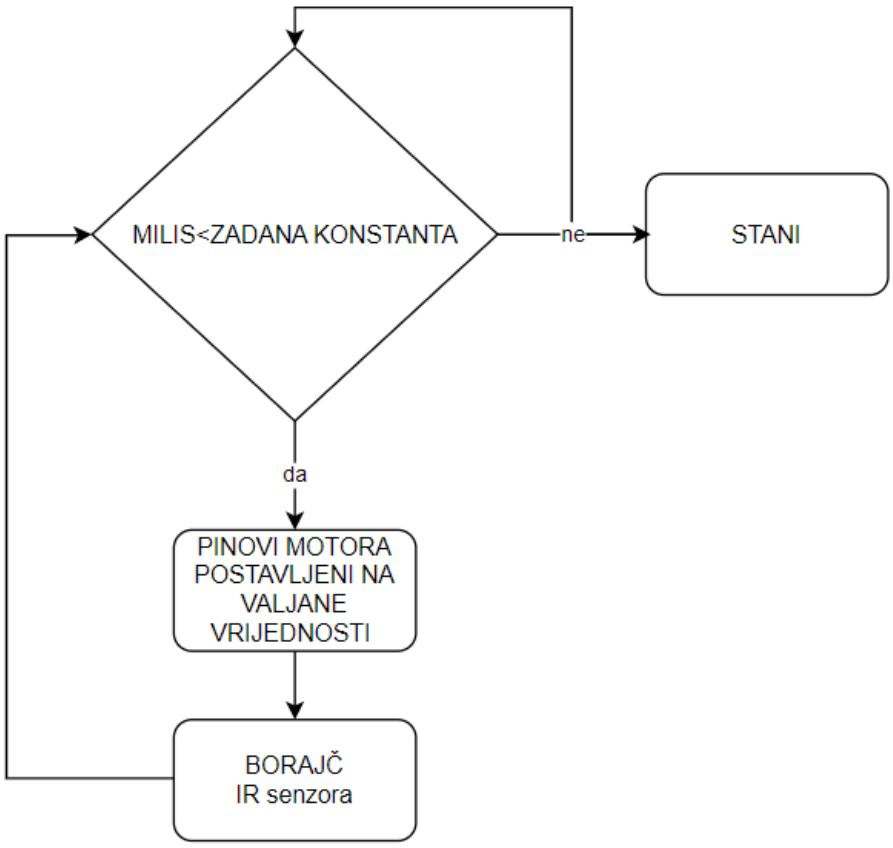
3.3.1.5. Izbjegavanje prepreka

Sijedeći dijagram predstavlja izbjegavanje prepreke. Prvo, funkcijom stop zaustavljeni su svi motori. Drugo, postavljena je pomoćna varijabla koja je bitna kako bi u programskom prekidu bilo moguće izračunati poziciju podnog čistača, a nakon toga postavljena je funkcija PWN, a s kojima se upravlja time koliki napon dobivaju motori. Ako je stavljeno 255, motor prima sav mogući napon kojega mogu dobiti putem motor driver-*a*, no ako je stavljen manji broj, primjerice samo postotak tog napona podni čistač će se kretati sporije. Brzina je bila bitna kako bi čišćenje metlice bilo temeljitije, kako bi IR senzori prepoznali svaku promjenu te kako bi bila smanjena mogućnost

proklizavanja na određenim podnim strukturama. *Delay* je služio kako bi se ogradili od proklizavanja te kako bi pričekali zaustavljanje podnog čistača. Za razliku od planiranog tijeka programa u dijelu 2.3., skretanje je vršeno unazad, a ne unaprijed kako bi se smanjila mogućnost kolizije podnog čistača sa prostornim preprekama. Nakon zaustavljanja *i delay-a* slučajnim odbairom pozivana je jedna od funkcija *goLeft* ili *goRight* koje skreću na način da se jedan od dva kotača vrti unatraške odnosno suprotno nego što bi se okretao u funkciji *goForward* te je na taj način dobiveno kružno skretanje prema nazad. Vremenski period koliko će skretati podni čistač unazad određen je s obzirom na PWN regulaciju motora kako bi podni čistač skrenuo za više od 90 stupnjeva te smanjio mogućnost kolizije s preprekama. Nakon kružnog skretanja unazad ponovo je pozvana funkcija *stop* te promijenjena vrijednost pomoćne varijable jer se podni čistač neće više kretati unazad. Ta varijabla detaljnije je objašnjena u dijelu 3.3.1.3.



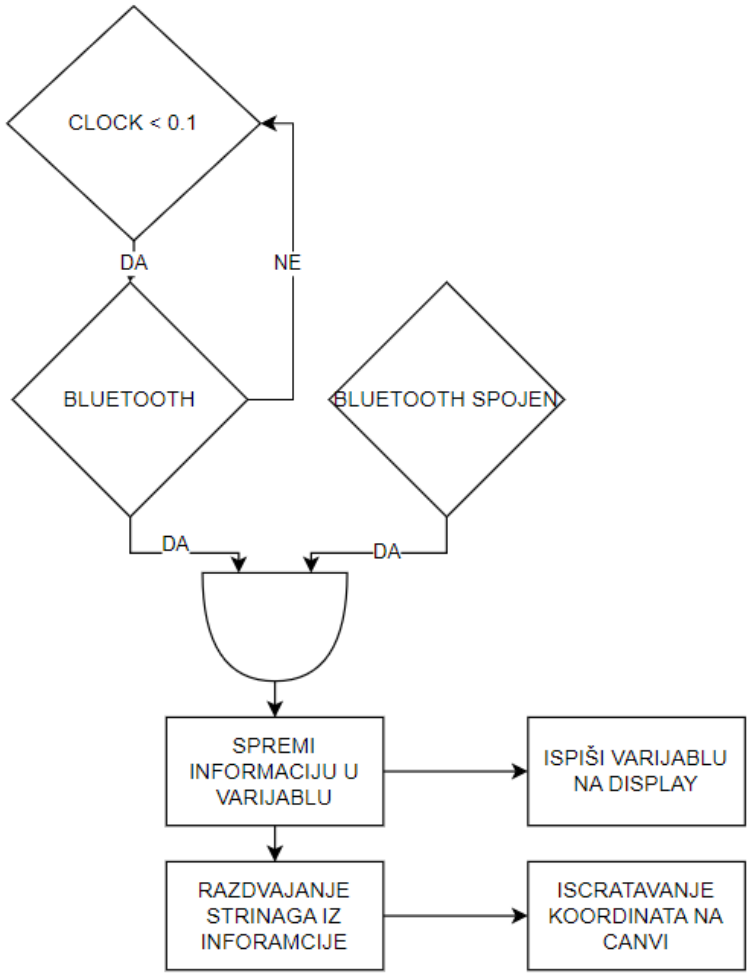
Slika 3.14. *Detaljan dijagram izbjegavanja prepreka.*



Slika 3.15. *Dijagram skretanja unazad, lijevo odnosno desno* .

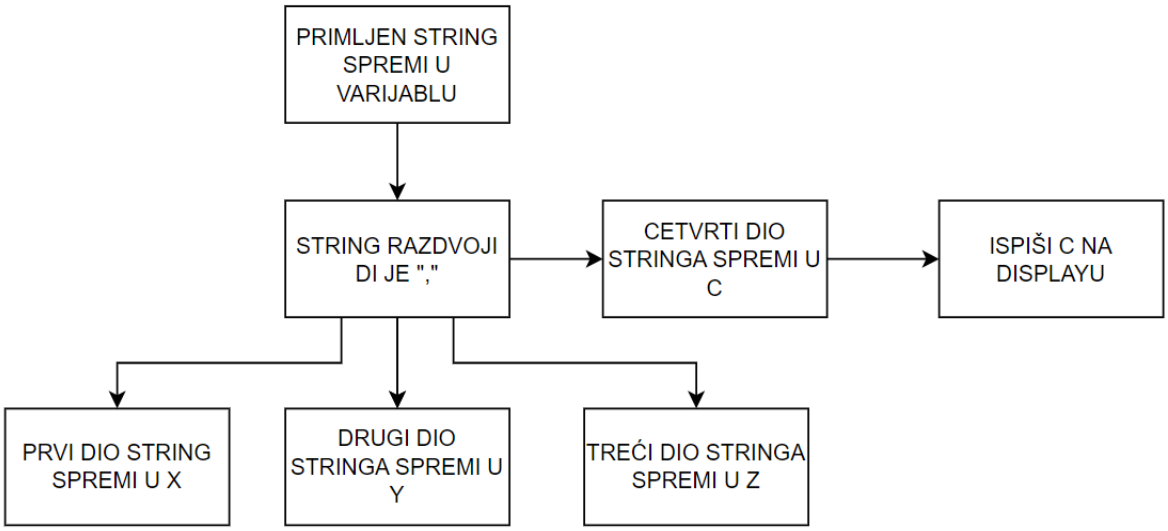
3.3.2. Realizacija aplikacijskog problemskog rješenja

Clock je svakih 0.1 sekundu provjeravao postoji li neka informacija za preuzimanje putem *bluetooth*- a. Ako ima i ako je ista imala više bitova od nula spremljena je u varijablu te ispisana na xyzc *label*-u koji ispisuje izvornu odnosno primljenu poruku. Poruka je tada poslana na dio gdje se iz *string-a* razdvajaju bitne informacije kako bi bilo moguće iscrtavanje grafa. Više o ovom dijelu u nastavku teksta.



Slika 3.16. *Blok dijagram aplikacijskog toka programa.*

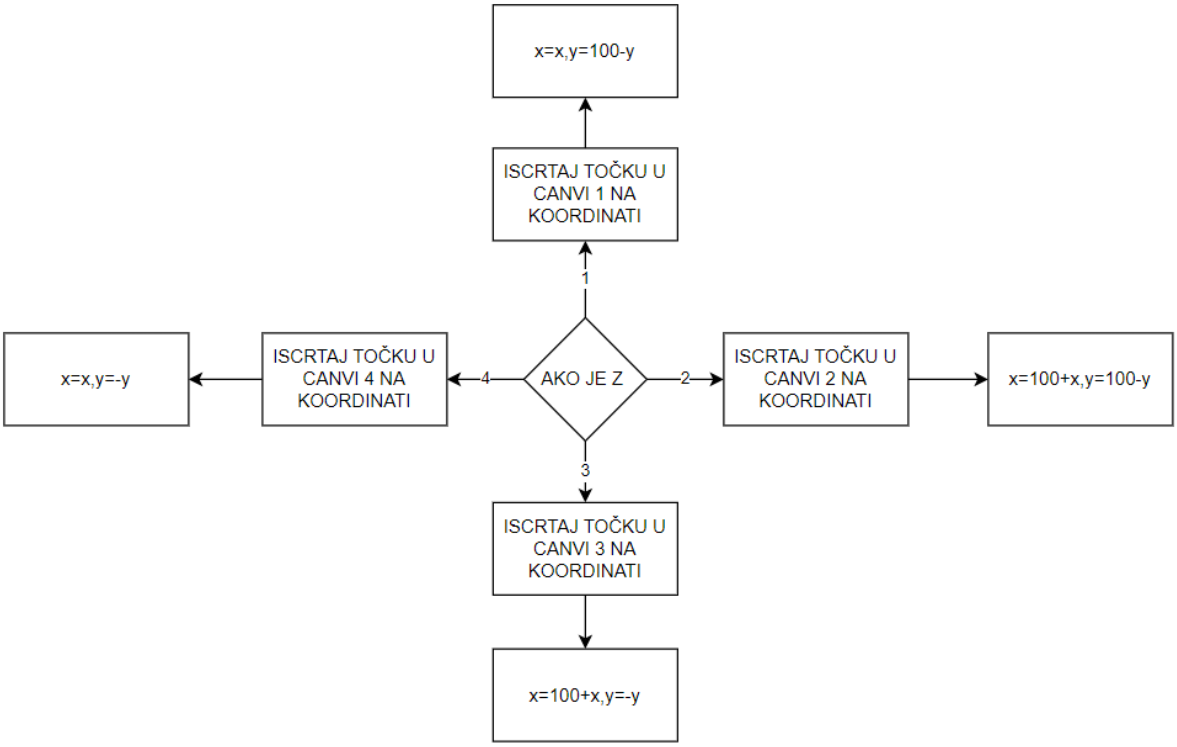
Na slijedećem dijagramu je prikazan način seciranja spremanja varijable u informacije koje su prikačene na display ili korištene u druge svrhe kao sto su određivanje koordinata ili *canve* koju treba koristiti.



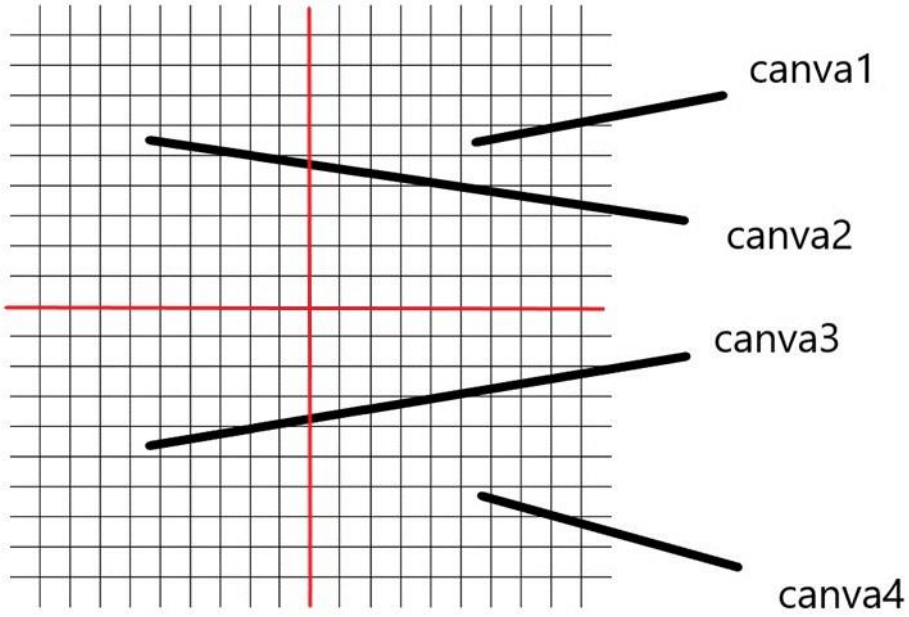
Slika 3.17. *Blok dijagram razdvajanja teksta odnosno stringa.*

X, y graf je složen pomoću četiri *canve* od kojih svaka čini jedan kvadrant istog grafa. *Canvine* koordinate se uvijek protežu od lijevog gornjeg vrha počevši sa nulom prema kraju desno odnosno dolje sa najvećom vrijednosti od 100 te osim određivanja u kojem kvadrantu je podni čistač odnosno u kojoj *canvi* se iscrtava potrebno je na dijagramu prikazane formule izvesti.

Trebalo je posebno obratiti pozornost na to da i x i y mogu doći kao negativne vrijednosti u određenim kvadrantima kao i smjer *canvinih* koordinata.



Slika 3.18. *Blok dijagram iscrtavanja koordinate s obzirom na canvu („z“).*



Slika 3.19. P*rikaz x,y grafa napravljen pomoću Canvi u aplikaciji.*

4. TESTIRANJE I REZULTATI

4.1. Metodologija testiranja

Nakon sklapanja hardvera te ispitivanja istog također je provjerena konfiguracija komponenata u programu. Potrebno je testirati dijelove, procese i rad istog kako bismo bili uvjereni u njegovu ispravnost. Metodologija rada podnog čistača je usko povezana sa gore navedenim problemima, a to su određivanje brzine kretanja i prepoznavanje objekta i izbjegavanje istih, praćenje trenutne pozicije.

Prepoznavanje objekta te izbjegavanje istih provjereno je puštanjem podnog čistača u određen prostor te brojanjem uspješno izbjegnutih kolizija s objektima u istom prostoru.

Praćenje trenutne pozicije testirano je puštanjem podnog čistača u rad te nakon prvog zaustavljanja uspoređena je njegova stvarna pozicija s pozicijom prikazanom na aplikaciji.

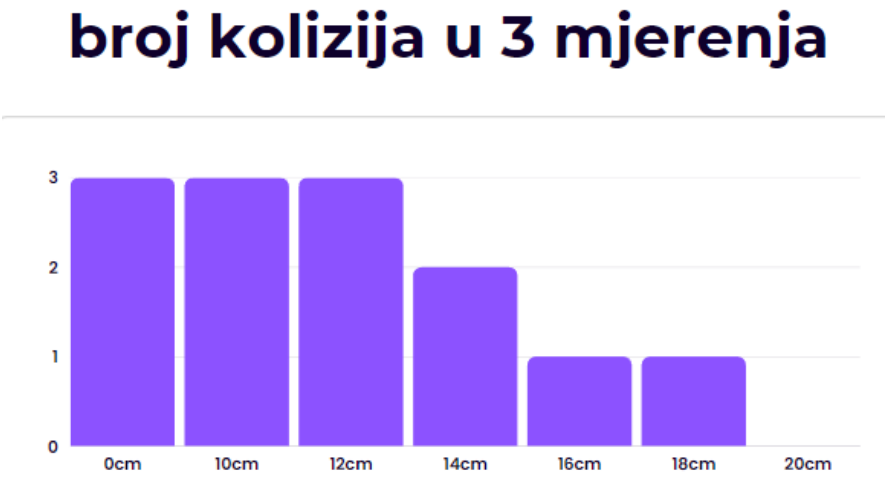
4.2. Rezultati testiranja

4.2.1. Prepoznavanje objekta te izbjegavanje istih

Slika 4.1. *Testiranje rada podnog čistača.*

Tablica 0.1. *Prikaz mjerenja stvarne zaustavne udaljenosti s obzirom na zaustavnu vrijednost unesenu u kod*

Vrijednost unesena u senzor	0cm	0 cm	0 cm	10cm	10cm	10cm	12cm	12cm	12cm
Udaljenost Čistača od prepreke nakon zaustavljanja	0cm	0cm	0cm	0cm	0cm	0cm	0cm	0cm	0cm
Vrijednost unesena u senzor	14cm	14cm	14cm	16cm	16.cm	16.cm	18cm	18cm	18cm
Udaljenost čistača od prepreke nakon zaustavljanja	0cm	0cm	1cm	6cm	1cm	0cm	5cm	3.5cm	0cm
Vrijednost unesena u senzor	20cm	20.cm	20cm						
Udaljenost čistača od prepreke nakon zaustavljanja	9.5cm	2.5cm	8.7cm						



Slika 4.2. *Grafički prikaz broja kolizija u 3 mjerenja.*

Slika 4.3. *Mjerenje udaljenosti prilikom testiranja.*

Procijenjeno je nakon višestrukog mjerenja da je najoptimalnija udaljenost na kojoj se čistač treba početi zaustavljati 20cm. Ta procjena proizlazi iz omjera unesene vrijednosti i udaljenosti zaustavljanja. Gore navedena vrijednost je odabrana jer takav omjer ima najveću vrijednost bez ijedne kolizije. Testiranje je bilo moguće nastaviti kako bismo bili u mogućnosti odabranu vrijednost usporediti s drugima, no nismo to nije bilo učinjeno jer je smatrano da bi veće unesene vrijednosti dovele samo do veće prosječne udaljenosti čistača nakon stajanja u slučaju da nije imao koliziju, a ta udaljenost nije od koristi jer se podni čistač neće dovoljno primaknuti prepreci i tako neće očistiti dovoljno blizu iste.

4.2.2. Usporedba stvarne pozicije s prikazanom na aplikaciji

Tablica 4.2. *Prikaz mjerenja prikazane pozicije na aplikaciji u odnosu na stvarnu.*

Broj mjerenja	Ručno izmjeren x (u cm)	Ručno izmjeren y (u cm)	X očitan iz aplikacije (u cm)	Y očitan iz aplikacije (u cm)	Razlika između očitanog i ručno izmjerenog x-a (u cm)	Razlika između očitanog i ručno izmjerenog y-a (u cm)
1.	-9.2	82.6	28.3	84.9	37.5	2.3
2.	-8.8	82.8	17.4	86.4	26.2	3.6
3.	-8.9	85.4	16.0	88.0	24.9	2.6
4.	-9.5	81.8	20.9	89.6	30.4	7.8
5.	-9.1	82.4	37.1	84.9	46.2	2.5
6.	-14.1	81.4	-12.6	90.0	1.5	7.6
7.	-9.2	83.4	-13.0	83.0	-3.8	-0.4
8.	-13.9	80.1	-14.1	80.4	-0.2	0.3
9.	-14.7	81.4	5.3	95.2	20.0	13.8
10.	-9.3	80.0	8.0	87.0	17.3	7.0
11.	-5.2	82.4	-4.5	86.1	0.7	3.7
12.	-9.4	81.6	-0.3	86.1	9.1	4.5
13.	-12	80.8	9.8	86.8	21.7	6.0
14.	-9.9	80.1	9.8	90.7	19.7	10.6
15.	-9.7	80.9	8.7	90.0	18.4	9.1
16.	4.2	80.0	32.2	77.5	28.0	-2.5
17.	3.2	79.0	50.0	72.5	46.8	-6.5
18.	2.8	78.4	6.2	88.8	4.4	10.4
19.	3.6	79.4	30.9	79.9	27.3	0.5
20.	4.9	83.2	12.4	86.2	7.5	3.0
21.	4.4	78.1	20.3	82.6	15.9	4.5
Suma	-129.5	1706.2	268.8	1796.6	399.5	90.4
Aritmetička sredina	-6.17	81.25	12.8	85.55	19.02	4.30

$$y_{ria} = \frac{\sum_{n=1}^{n} y_{rin}}{n} \quad (4\text{-}2)$$

$$x_{oa} = \frac{\sum_{n=1}^{n} x_{on}}{n} \quad (4\text{-}3)$$

$$y_{oa} = \frac{\sum_{n=1}^{n} y_{on}}{n} \quad (4\text{-}4)$$

$$x_r = x_o - x_{ri} \quad (4\text{-}5)$$

$$y_r = y_o - y_{ri} \quad (4\text{-}6)$$

- x_{ri}– ručno izmjeren pomak s obzirom na koordinatnu os po x osi,

- y_{ri}– ručno izmjeren pomak s obzirom na koordinatnu os po y osi,

- x_o– pomak očitan iz aplikacije s obzirom na koordinatnu os po x osi,

- y_o– pomak očitan iz aplikacije s obzirom na koordinatnu os po y osi,∑

- x_r– razlika između očitanog i ručno izmjerenog x-a

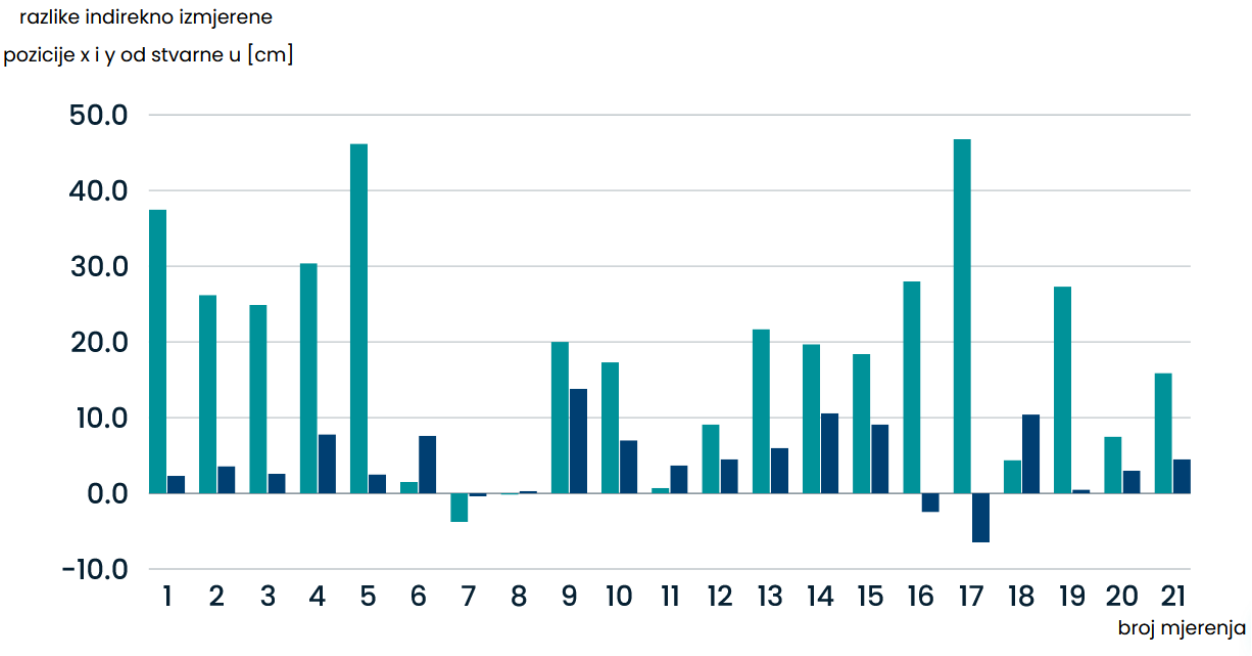
- y_r– razlika između očitanog i ručno izmjerenog y-a

-x_{ria} – aritmetička sredina od ručno izmjerenog x

-y_{ria} – aritmetička sredina od ručno izmjerenog y

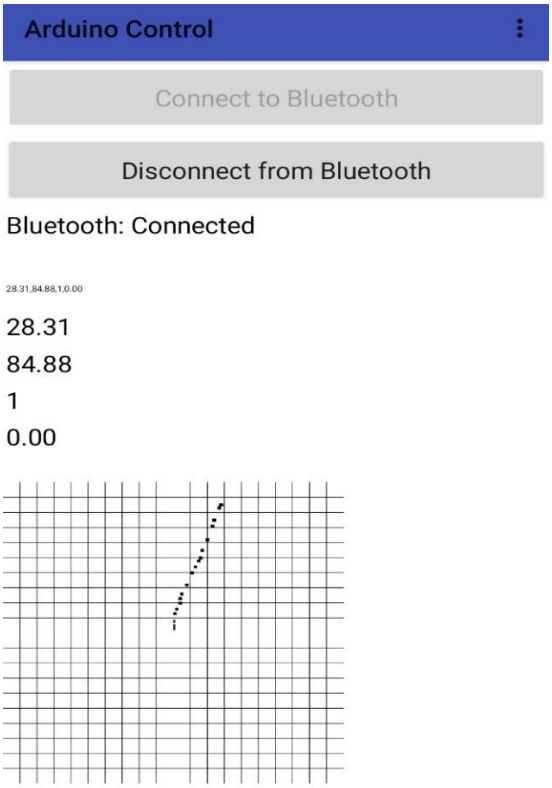
-x_{oa} – aritmetička sredina od očitanog x

-y_{oa} – aritmetička sredina od očitanog y



Slika 4.4. *Grafički prikaz razlike indirektno izmjerene pozicije x i y od stvarne.*

Prilikom mjerenja utvrđene su konstante greške kako su X i Y u prosjeku veći od stvarnog. Oscilacije su valovitog oblika zbog nemoguće kalibracije potenciometra koja proizlazi iz preosjetljivosti potenciometra za kalibraciju IR senzora. Nakon 5., 10. mjerenja vršena je kalibracija IR senzora. Jasno je vidljivo kako nakon svake kalibracije senzora pomoću njegovog potenciometra, prvo mjerenje odnosno 6. i 11. su najtočniji stvarnom X-u i Y-u, dok se u sljedećim mjerenjima greška počinje ponovno oscilirati. Nakon 15. mjerenja vršena je kalibracija PWN regulacije motora dobivene su i dalje velike oscilacije rješenja, ponajviše u desno odnosno po X osi kao i u prijašnjim mjerenjima. Lako je zaključiti da je IR senzor na oba kotača, ponajviše na lijevom, stvarao probleme koje je bilo moguće riješiti na način da je ostavljen IR senzor s konstantom istom greškom te pomnožen u kodu izračun pozicioniranja s određenom konstantom koja smanjuje grešku IR senzora. No, ni to ne bi u potpunosti riješilo problem jer greške previše osciliraju.



Slika 4.5. S*lika aplikacijskog prikaza testiranja.*

Slika 4.6. *Prikaz testiranja.*

5. ZAKLJUČAK

Napredak u tehnologiji uvelike je olakšao živote ljudima koje su na većini industrijskih poslova nakon dugo vremena počeli zamjenjivati strojevi. Sustav za čišćenje podova mali je kućni robot koji radi pomoću senzora i drugih robotskih dijelova pri čemu cjelokupni posao obavlja prema unaprijed osmišljenom algoritmu. Senzori su tako najbitniji dio svakog robotskog usisavača jer su upravo oni ti koji uočavaju promjene i zapreke u okolini čistača, pri čemu daju određene izlazne signale koji se onda šalju u mikroupravljač koji „odlučuje“ o daljnjim kretnjama u radu sustava.

Ovim radom htjelo se saznati nešto više o zahtjevima i radu jednoga sustava za čišćenje podova za što je također bilo potrebno izraditi i jedan model takvoga sustava. Taj je model morao imati i funkciju detekcije i izbjegavanja prepreka koje mu se nađu na prostoru čišćenja, izračun trenutne pozicije, pohrana tih podataka u memoriju te slanje tih info na aplikaciju na kojoj je iscrtavana u koordinatnom sustavu trenutna pozicija podnog čistača. Kao prijedlog sklopovskog rješenja ponuđen je oblik valjka koji bi bio najbolje rješenje za ovaj zadatak. Pri tome, model ima otvore za baterije i motor, kotače, četku i dvije sklopke. Za programsko rješenje sustava korišten je programski alat ArduinoIDE i *MitApp Inventor* za aplikacijski dio završnoga rada. Prije svega odabran je materijal, a nakon toga izrađeni su dijelovi 3D printerom. Nakon toga postavljene su komponente na za to predviđena mjesta na bazi, pri čemu je došao do izražaja problem nedostatka mjesta za prekidače. Također, stvoren je problem kako izračunati u cm koliko je podni čistač prešao puta. Taj je problem riješen tako što IR senzor prepozna promjenu stanja koje nastupa svaki puta kada prepozna ili prestaje prepoznati kostur kotača te se množi s određenim koeficijentom koji ovisi o opsegu istoga kotača kako bi dobili pređen put. Slijedeći problem koji se pojavio bio je dok se izvršava kretanje naprijed ili skretanje unazad paralelno s tim nije bilo moguće provjeravati promjenu stanja na IR senzoru. To je riješeno tako što nije korišten *delay* nego funkcija millis koja na određeno vremensko razdoblje poziva funkcije koje ne samo da pokreću motore nego i pozivaju funkciju koja provjerava IR senzore. Problem zapisivanja trenutne pozicije podnog čistača riješen je na način da se prilikom početka kretanja podnog čistača smatra da uvijek ide prema sjeveru s obzirom na x i y osi. Pomoću znanja broja promjena stanja na IR senzorima iz kojih se izračunava pređen put i razlika prijeđenog puta između izračunavan je pređen put i promjenu smjera. Prilikom skretanja unazad oduzimala se promjena pozicije od posljednje pozicije kako bi bila dobivena stvarna pozicija. Izračunavanje prosječne brzine kretanja je

izvedeno tako što znamo prosječan put svakog programskog prekida i znajući da je isti pozvan svakih 0,1s dobivena je vrijednost brzine. Spremanje koordinata u memoriju koja se resetira ponovnim korištenjem izvedeno je pomoću NVS memorije i brojača koji služi kako koordinate ne bi bile pisane na jedan te isti blok odnosno kako bi svaka prijašnja koordinata bila spremljena na svoju poziciju. Brisanje memorije izvedeno je prilikom uključivanje podnog čistača kada se briše NVS memorija od prošlog korištenja. Nakon postavljanja, uslijedilo je spajanje komponenti i provjera rada sustava. Tijekom testiranja rada ustanovljeno je kako sustav radi prebrzo, kako bi IR senzori mogli prepoznati svaku promjenu stanja, pa je taj problem riješen pomoću PWN regulacije. Osim toga, provedenim testiranjima zaključeno je koja je udaljenost na kojoj će se sustav početi zaustavljati.

Iz testiranja je vidljivo kako rezultati zaustavljanja od prepreke previše osciliraju zbog načina funkcioniranja ultrazvučnog senzora i zbog njegovog malog kuta detekcije, jedno rješenje tog problema bilo bi korištenje više ultrazvučnih senzora kako bi se povećao kut detekcije te onda čistač ne bi zapinjao kotačima za rubove prepreka. Također je utvrđeno da čistač doživljava koliziju kada prilazi prepreci koja je pod kutom zbog načina funkcioniranja UZ senzora, odnosno zbog ne vraćanja *echo*-a od prepreke. Taj problem je rješiv promjenom UZ senzora s ultracrvenim senzorom ili *sweeper*-om. Ustanovljeno je i da bi trebalo postaviti i određenu zaštitu od pretjeranog prašenja s donje strane čistača kako se prašina i prljavština ne bi nahvatale na komponente čistača. Jedan od problema čistača je prijelaz na otirač zbog krajeva četke koji su kruti i pozicionirani nisko zbog čega zapinju za otirač. Kako bi se riješio taj problem potrebno je pronaći četku s labavijim krajevima, ali u tom slučaju postoji problem slabijeg prianjanja na površinu odnosno lošijeg čišćenja. Bilo bi efikasnije da je korištena jedna baterija drugačijih specifikacija kako bi se uštedilo na prostoru i smanjila kompleksnost samog sklopa. Jedan od problema bio je i slanje i spajanje s aplikacijom što je riješeno sa uvjetom „*if*“ koji ne omogućava početak rada podnog čistača ako nije spojena na *bluetooth*. u aplikacijskom dijelu najveći problem bio je iscrtavanje na grafu pozicije podnog čistača. Taj se problem pokušao riješiti tako što su putem WiFi-ja s ESP32 slane koordinate i brzine na server od *ThinkSpeake*-a. U *ThinkSpeake* pomoću *Mathlaba* isprogramiran je graf na kojemu bi se iste koordinate iscrtavale te link istog x,y grafa je postavljen u *Mitapp invento*r aplikaciju. Ta ideja nije bila odgovarajuća jer je informacija kasnila dvije sekunde pa je pokušano drugo programsko rješenje, a to je da su 4 *canve* spojene u jedan veliki pravokutnik gdje svaka predstavlja jedan kvadrant. Te 4 *canve* su predstavljale problem jer

je potrebno znati u kojoj *canvi* iscrtavati koordinatu. Taj je problem riješen izračuna u kojem se kvadrantu nalazi koordinata prilikom slanja u programskom prekidu i onda je ta info poslana putem *bluetooth*-a na *MitApp* aplikaciju. Problem kod *canvi* također je što njihove koordinate idu od 0 iz gornjeg lijevog kuta do 100 u donji desni kut. Osnovnim matematičkim funkcijama u *MitApp Inventoru* napravljeno je iscrtavanje te koordinate na način da je u središtu gdje se diraju sve 4 *canve* 0,0 koordinata. Ovo rješenje ima problem što može mapirati samo 2x2m prostor. To je rješivo na način da se prilikom slanja informacije putem *bluetooth* aplikacije podijele x i y s određenom konstantom kako bi se taj prostor multiplicirao za tu konstantu. Problem sa rastavljanjem informacije na 4 različite koje su primane sa mobilnog čistača je riješen tako da su x,y,z, i c razdvojeni zarezom kojega je aplikacija prepoznala te na temelju njega može razvrstala te poslane varijable. U radu je u postojanju i problem s bitovima koji nije riješen. Kod je napravljen tako da je moguće primiti *string* od neodređenog broja bitova, no i dalje se aplikacija rušila zbog greške kada *clock* prepozna promjenu u *stringu,* a novi *string* nije poslan te je veličina *stringa* 0 bitova i tada dolazi do izlaska upozorenja u aplikaciji. Klikom pored upozorenja srušene aplikacije, ignorirana je ta greška te aplikacija normalno nastavlja raditi.

Prilikom testiranja i puštanja u rad stvarna pozicija podnog čistača za dosta oscilira od prikazane na aplikaciji zbog IR senzora kojega je gotovo nemoguće kalibrirati zbog prevelike osjetljivosti njegovog potenciometra. Zbog iste osjetljivosti problem je pokušan riješiti usmjeravanjem LE dioda koje šalju i primaju signal na IR senzoru no ni on nije doveo do rješenja. Iz testiranja se može zaključiti da nakon kalibracije potenciometra se na kratko vrijeme povećava točnost pozicije no nakon određenog vremena IR senzor lijevi, a ponajviše desni, smanjuje prepoznavanje promjena. Ovaj problem moguće je riješiti izostavljanjem kalibracije IR senzora i izračunavanjem njihove prosječne greške te pomoću nje u kodu ispravljanjem iste.

6. LITERATURA

[1] Tapio Taipalus, Using Remote Controlled Service Robot for Fetching Objects in Home . Environment. 2004, ResearchGate.

https://www.researchgate.net/publication/320672889_Using_Remote_Controlled_Service_Robot_for_Fetching_Objects_in_Home_Environment/link/59f341acaca272607e2904e3/download [pristup 27.6.2023.]

[2] Rajaranjan Senapati, Automation and controlling of automatic floor cleaner. 2014. A bachelor thesis. Department of Mechanical Engineering. National Institute of Technology Roukela Orissa-769008, India.

http://ethesis.nitrkl.ac.in/7500/1/147.pdf [pristup 27.6.2023.]

[3] Pernilla Sanden, Jeff Pertot, Autonomous cleaning robot as a service, 2020, MA thesis. Linköping's university.

https://www.diva-portal.org/smash/get/diva2:1444501/FULLTEXT01.pdf [pristup 27.6.2023.]

[4] Lea Mets Kiritsis. Can cheap robotic vacuum cleaners be made more efficient? 2018, Bachelor Thesis, Kungliga Tekniska H¨ogskolan, OAI: DiVA.org:kth-230228 [pristup 28.8.2023.]

[5] Alberto Rodriguez, High-resolution tactile sensing for reactive robotic manipulation. 2021. https://dspace.mit.edu/handle/1721.1/130764 [pristup 28.8.2023.]

[6] Table 8. The advantages and disadvantages of MIT App Inventor (N=36). (n.d.). ResearchGate. https://www.researchgate.net/figure/The-advantages-and-disadvantages-of-MIT-App-Inventor-N36_tbl5_281847214 [pristup 28.8.2023.]

[7] L298N Motor Driver Bord

https://www.geeetech.com/wiki/index.php/L298N_Motor_Driver_Board?fbclid=IwAR1LxUoUdKUBeZDoOxKTgroR1fVy5CDUqRCS0T1lWAwvcFCNY9m4EarYRAY [pristup 27.6..2023.]

[8] Understanding how ultrasonic sensors work.

https://www.maxbotix.com/articles/how-ultrasonic-sensors-work.html [pristup 28.8.2023.]

[9] How HC-SR04 ultrasonic sensor works & Interface it with Arduino.

https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/ [pristup 28.8.2023.]

[10] L298N Motor Driver – Arduino Interface, How It Works, Codes, Schematics. https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/?utm_content=cmp-true [pristup 28.8.2023.]

[11] *DC Series Motor : Circuit Diagram, Characteristics and its Applications.* ElProCus - https://www.elprocus.com/dc-series-motor-components-circuit-diagram-applications/ [pristup 28.8.2023.]

[12] Infrared Sensors Overview: Types, Functioning and Use Cases.

https://www.getkisi.com/blog/infrared-sensors-overview-types-functioning-and-use-cases

[13] Danny Jost, What is an IR sensor? Fierce Electronics, 2019.

https://www.fierceelectronics.com/sensors/what-is-an-ir-sensor [pristup 28.3.2024.]

[14] Battery Skills. (n.d.). 1.2V vs. 1.5V Battery - Battery Skills.

https://www.batteryskills.com [pristup 28.3.2024.]

[15] Electricity Magnetism. (n.d.). 1.5V Battery | Type, Size & Characteristics. https://www.electricity-magnetism.org [pristup 28.3.2024.]

7. PRILOZI I DODACI

7.1 Hardverske komponente

7.1.1 L298N MOTOR DRIVER

L298N motor driver elektronički je sklop (dvostruki H-most) koji omogućuje kontrolu brzine i smjera dva istosmjerna motora u isto vrijeme. Takav modul može pokretati istosmjerne motore koji imaju napon između 5 i 35V, s najvišom strujom do 2A. Modul ima dva bloka s vijčanim stezaljkama za motor A i B, i još jedan blok s vijčanim stezaljkama za pin za uzemljenje, VCC za motor i pin od 5V koji može biti ulaz ili izlaz [7].

7.1.2. ESP32-WROOM-32

ESP32-WROOM-32 je snažan, generički modul koji cilja na široku raspon raznih primjena, u rasponu od najmanjih zadataka za senzore male snage do najzahtjevnijih zadataka, poput kodiranja glasa, streaminga glazbe i MP3 dekodiranja. Srž ovog modula je ESP32-D0WDQ6 čip [7]. Ugrađeni čip dizajniran je da bude skalabilan i adaptivan. Postoje dvije CPU jezgre koje se mogu zasebno kontrolirati, a frekvencija CPU takta je podesiva od 80 MHz do 240 MHz [7]. Čip također ima koprocesor male snage koji se može koristiti umjesto CPU-a i tako štedjeti energiju dok obavljamo zadatke koji ne zahtijevaju mnogo računalne snage [7]. Integracija Bluetootha, Bluetooth LE i Wi-Fi-ja osigurava ciljanje na širok raspon aplikacija, te da je modul sveobuhvatan: korištenje Wi-Fi-ja omogućuje veliki fizički domet i izravnu vezu s internetom putem Wi-Fi usmjerivača, dok korištenje Bluetootha omogućuje korisniku praktično povezivanje telefonom [7].

7.1.3. ULTRAZVUČNI SENZOR ARDUINO HC- SR04

Ultrazvučni senzor HC-SR04 koristi SONAR za određivanje udaljenosti objekta baš kao što to rade šišmiši. Nudi izvrsnu beskontaktnu detekciju dometa s visokom preciznošću i stabilnim očitanjima u pakiranju jednostavnom za korištenje od 2 cm do 400 cm ili 1 do 13 stopa [9]. Na rad ne utječu sunčeva svjetlost ili crni materijal, iako akustički, meke materijale poput tkanine može biti teško otkriti [9]. Dolazi u kompletu s modulom ultrazvučnog odašiljača i prijemnika. Glavne tehničke specifikacije su mu napajanje od +5V DC, struja mirovanja <2mA, radna struja 15mA, efektivni kut <15°, udaljenost dometa 2 cm – 400 cm/1″ – 13 stopa, razlučivost od 0,3 cm i mjerni kut − 30 stupnjeva [10]. Ultrazvučni senzor udaljenosti HC-SR04 zapravo se sastoji od dvije ultrazvučne sonde. Jedna djeluje kao odašiljač koji pretvara električni signal u ultrazvučne zvučne impulse od 40 kHz.

Druga djeluje kao prijemnik i osluškuje odaslane impulse [10]. Kada prijemnik primi te impulse, proizvodi izlazni impuls čija je širina proporcionalna udaljenosti objekta ispred [10]. Budući da radi na 5 volti, može se spojiti izravno na Arduino ili bilo koji drugi logički mikroupravljača od 5 V [10]. Sve počinje kada je *pin* okidača postavljen na HIGH na 10µs. Kao odgovor, senzor odašilje ultrazvučni niz od osam impulsa na 40 kHz. Ovaj 8-pulsni uzorak posebno je dizajniran tako da prijamnik može razlikovati odaslane impulse od ambijentalnog ultrazvučnog šuma [11]. Ovih osam ultrazvučnih impulsa putuje kroz zrak dalje od odašiljača. U međuvremenu *echo pin* postaje HIGH kako bi pokrenuo povratni signal *echo*-a. Ako se ti impulsi ne reflektiraju natrag, signal eho-a istječe i postaje niski nakon 38ms (38 milisekundi*)* [11]. Stoga puls od 38 ms ukazuje da nema prepreka u dometu senzora. Ako se ti impulsi reflektiraju natrag, *echo pin* postaje niski čim se signal primi [11]. Ovo generira impuls na *echo pin-u* čija širina varira od 150 µs do 25 ms ovisno o vremenu potrebnom za prijem signala [11].

7.1.4. 9V PUNJIVA BATERIJA

9V baterije su jedne od vrlo traženih baterija zbog mnogih aplikacija ili uređaja koje mogu napajati. Svaka baterija od 9 volti doslovno je opisana kao ćelija s pravokutnom prizmom sa zaobljenim rubovima i polariziranim uskočnim konektorom na vrhu [7]. Ima dimenzije proizvoda od približno 46,40 mm do 48,50 mm visine; 25,0 mm do 26,50 mm približne duljine i širine u rasponu od 15,0 mm do 17,50 mm [7]. Prvo se koristi u tranzistorskim radijima. To je razlog zašto neki proizvođači

baterije od devet volti još uvijek nazivaju tranzistorskim baterijama [7]. Baterija od 9V dolazi u dvije osnovne vrste. To su primarne 9V baterije i 9-voltne punjive baterije. Primarni imaju različitu kemiju ili sastav [7]. Punjive baterije također su izrađene od različitih kemijskih sastava s malim razlikama u nekim značajkama. Primarne ćelije su ili alkalne baterije, litij, cink-zrak, cink klorid, srebrov oksid i druge varijante [7]. S druge strane, punjive 9-voltne baterije sastoje se od takvih kemikalija kao što su punjive alkalne baterije, litij-ion, litij željezo fosfat (LiFePO4), litij mangan oksid (LiMn2O4), litij titanat (Li2TiO3), litij nikal mangan kobalt oksid (LiNiMnCoO2), litij kobalt oksid (LiCoO2), nikal kadmij (NiCad), nikal metal hidrid (NiMH), olovna kiselina [7].

Različite kemije imaju različite karakteristike. To utječe na izvedbu i kompatibilnost s odabranim uređajima. Punjive baterije od devet volti obično dolaze u dvije opcije: to su NiMH i litijeve [7]. Litijske baterije udvostručuju NiMH kapacitet [7]. Kapacitet NiMH baterije od 9 V traje oko četiri sata s punjivom litijskom baterijom od 7 do 7,5 sati. Punjiva baterija od 9 V dolazi u dvije varijante veličine [7]. Ovo su PP3 i PP9. Postoje varijacije veličine punjive baterije od devet volti kako bi se omogućio veći kapacitet punjenja, kad god je to potrebno [7]. Baterija od devet volti, jednokratna ili punjiva, obično se koristi u dimnim alarmima, detektorima dima, walkie-talkiejima, tranzistorskim radijima, testnim i instrumentacijskim uređajima, medicinskim baterijama, LCD zaslonima i drugim malim prijenosnim uređajima [7]. Baterije od devet volti također se idealno koriste u monitorima pacijenata, uređajima za nadzor, svjetionicima za hitne slučajeve, snimačima podataka, kirurškoj rasvjeti i drugim medicinskim uređajima, opremi i instrumentima [7]. 9V se često ne koriste samo u zdravstvenoj industriji, već su također popularni u proizvodnji, komercijalnim nekretninama, obrazovanju, ugostiteljstvu ili bilo kojem drugom sektoru. 9V punjive baterije obično dolaze s posebnim punjačem jer oni nisu kompatibilni s univerzalnim punjačima ili bilo kojim drugim punjačem baterija [7]. Većina 9V baterija prodaje se na tržištu zajedno s odgovarajućim punjačima baterija u pakiranju [7]. Jedna od prednosti nabave ovih baterijskih paketa s punjačima je ta što vam mogu uštedjeti vrijeme i pomoći u optimizaciji vaših baterija, kao i performansi vaših uređaja ili opreme budući da su to kompatibilni proizvodi [7].

7.1.5. ISTOSMJERNI MOTOR

DC istosmjerni motor sličan je bilo kojem drugom motoru jer je glavna funkcija ovog motora pretvaranje električne energije u mehaničku energiju [12]. Rad ovog motora uglavnom ovisi o elektromagnetskom principu [12]. Kad god se magnetsko polje približno formira, vodič kroz koji prolazi struja surađuje s vanjskim magnetskim poljem, a zatim se može generirati rotacijsko gibanje [12]. Komponente ovog motora uglavnom uključuju rotor (armaturu), komutator, stator, osovinu, namotaje polja i četke. Fiksna komponenta motora je stator, a građen je od dva elektromagnetska pola[12]. Rotor uključuje armaturu i namote na jezgri povezane s komutatorom. Izvor energije može se spojiti prema namotima armature kroz niz četkica povezanih s komutatorom [12].

Rotor uključuje središnju osovinu za rotaciju, a namot polja mora biti u stanju držati veliku struju zbog veće količine struje u cijelom namotu, što će veći biti okretni moment proizveden s motorom [12]. Stoga se namot motora može izraditi od pune žice [12]. Ova žica ne dopušta veliki broj zavoja.

Namot se može izraditi od čvrstih bakrenih šipki jer pomaže u jednostavnom, ali i učinkovitom odvođenju topline koju stvara velika količina struje tijekom namotavanja [12]. Kontrola brzine istosmjernog motora može se postići korištenjem dvije sljedeće metode:

- metoda kontrole protoka
- metoda kontrole otpora armature.

Najčešće korištena metoda je metoda kontrole otpora armature [12]. U metodi kontrole otpora armature, promjenjivi otpor se može izravno spojiti u seriju kroz napajanje [12]. Ovo može smanjiti napon koji je dostupan putem armature i pad brzine. Promjenom vrijednosti promjenjivog otpora može se postići bilo koja brzina ispod redovite brzine [12]. Ovo je najopćenitija metoda koja se koristi za kontrolu brzine istosmjernog motora.

Prednosti DC serijskog motora uključuju sljedeće:

- veliki startni moment
- jednostavna montaža i jednostavan dizajn
- zaštita je laka
- Isplativ.

Nedostaci DC serije motora uključuju sljedeće:

- regulacija brzine motora je prilično loša. Kada se brzina opterećenja povećava, tada će se brzina stroja smanjiti,
- kada se brzina poveća, moment DC motora će se naglo smanjiti. Ovaj motor uvijek treba opterećenje prije pokretanja motora. Stoga ovi motori nisu prikladni tamo gdje je opterećenje motora potpuno uklonjeno [12].

7.1.6. IR senzor

Infracrveni (IR) senzori su uređaji koji otkrivaju i mjere zračenje emitirano ili reflektirano od objekata. Sve što emitira zračenje je pod utjecajem topline. IR senzori se koriste na razne načine zbog svoje sposobnosti detekcije kretanja, mjerenja temperature i bilježenja promjena u okolini. IR kontroler ima široku primjenu u kućnim aplikacijama gdje se koristi za upravljanje raznim uređajima kao što su klima uređaji, televizori i rasvjeta putem daljinskog upravljača [12]. Koriste se u industriji za nadzor proizvodnje, otkrivanje požara i mjerenje temperature. U automobilima, IR senzori mogu pomoći sustavima za noćno gledanje i senzorima za parkiranje [12]. IR senzor također se koristi u sigurnosnim sustavima gdje može detektirati upade bez lažnih alarma koje tradicionalni detektori mogu izazvati [12]. Njegove primjene također uključuju zdravstvenu tehnologiju; Na primjer, koriste se u vatrogasnim kamerama za mjerenje tjelesne temperature kao sredstvo otkrivanja požara. Infracrveni senzori koriste se u mnogim uređajima i aplikacijama, kao što su daljinski upravljači za televizore i klima uređaje [13]. Oni detektiraju infracrveno zračenje koje emitiraju ili reflektiraju objekti, što ih čini prikladnim za praćenje promjena u okolini bez fizičkog kontakta. Neki infracrveni senzori dizajnirani su za otkrivanje topline koju emitira ljudsko tijelo, što ih čini korisnima u sigurnosnim sustavima i kućnoj automatizaciji [13]. Tehnologija pasivnog infracrvenog senzora (PIR) koristi se u alarmnim sustavima jer može otkriti kretanje na temelju promjena u infracrvenom zračenju okoline. U medicini se infracrveni senzori koriste u termoviziji za otkrivanje područja povišene tjelesne temperature, što može ukazivati na upalu ili infekciju [13]. Inovacije u tehnologiji infracrvenog senzora omogućuju njegovu upotrebu u mobilnim telefonima i igraćim konzolama za prepoznavanje gesta i praćenje pokreta. Osim toga,

infracrveni senzori ključni su za razvoj autonomnih vozila jer se koriste za otkrivanje objekata i navigaciju [13].

7.1.7. 1.5V baterija

Baterije od 1,5 V standardni su izvor napajanja u mnogim prijenosnim elektroničkim uređajima, zahvaljujući njihovom dosljednom izlazu i širokoj dostupnosti [14]. Ove baterije obično dolaze u uobičajenim veličinama kao što su AAA, AA i C ćelije i koriste se u raznim kućanskim i potrošačkim proizvodima [14]. Jedna od ključnih prednosti baterija od 1,5 V je njihova sposobnost da daju izlaz visokog napona, što je bitno za uređaje koji zahtijevaju više energije za učinkovit rad, kao što su digitalne kamere i prijenosni audio uređaji [15].

Postoji nekoliko vrsta baterija od 1,5 V, a alkalne su najčešće. Alkalne baterije nude veću gustoću energije i dulji vijek trajanja od mnogih drugih vrsta, što ih čini idealnim za širok raspon primjena [15]. Dostupne su i litijeve baterije od 1,5 V, koje pružaju još veću gustoću energije i prošireni radni raspon na visokim i niskim temperaturama [15]. Učinkovitost baterija od 1,5 V može varirati ovisno o njihovom kemijskom sastavu. Na primjer, litijeve i alkalne baterije ponašaju se različito u različitim uvjetima okoline, pri čemu litij obično ima bolje rezultate u hladnijim klimatskim uvjetima [15]. Važno je uzeti u obzir zahtjeve napona uređaja i kompatibilnost pri odabiru baterije kako biste osigurali optimalne performanse i izbjegli potencijalno oštećenje uređaja [15].

7.2 Program (kôd) završnoga rada

```
#include "BluetoothSerial.h"
BluetoothSerial SerialBT;
//interupt
#include "freertos/FreeRTOS.h"
#include "freertos/timers.h"
#define TIMER_PERIOD pdMS_TO_TICKS(100)
TimerHandle_t myTimer;
```

```
//memorija
#include <Preferences.h>
Preferences preferences;

// 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;
//PWM
int speed1, speed2, speed3;
int ena = 14;
int enb = 13;//21
// Ljevi motor
int leftMotorForwardPin = 17;
int leftMotorBackwardPin = 16;//BILO 17
// Desni motor
int rightMotorForwardPin = 32;
int rightMotorBackwardPin = 33;
// Definiranje pina na koji je spojen izlaz infracrvenog senzora
const int irPin1 = 27;  // Promijenite broj pina prema stvarnom spoju
const int irPin2 = 15;
int BrojPromjenaSatanja1 = 0;
int BrojPromjenaSatanja2 = 0;
int prethodnoStanje1 = LOW;
int prethodnoStanje2 = LOW;
int KonstantaUdaljenosti1= 1;// Potrbno izračunat mjerečiu kotac
int KonstantaUdaljenosti2= 1;// Potrbno izračunat mjerečiu kotac
int BrojKuta = 0;
```

```
int KonstantaKuta = 1;// Potrbno izračunat mjerečiu kut skretanja
int Kut = 0;
       int BrojPromjenaSatanja22 = 0;
            int BrojPromjenaSatanja11 = 0;
unsigned long previousMillis = 0;
unsigned long interval = 2000;
unsigned long interval2 = 5000;
  bool nazad = LOW;
  float razlika =0,razlika2 =0;
  float smjer = 0;
  float smjer_radijani = 0;
  float prosjek =0;
  float d=0, c=0, cc=0, ccc=0, dd=0;
  int z=0;
  float x=0, y=0,dx=0, dy=0;
// Ova funkcija će se pozvati kada timer istekne
void TimerCallback(TimerHandle_t xTimer) {
   razlika = (BrojPromjenaSatanja2 - BrojPromjenaSatanja1)*2.1;
   razlika2+=razlika;
   smjer = atan2(razlika, 26);
    prosjek = (BrojPromjenaSatanja1 + BrojPromjenaSatanja2) / 2.0;
    d = prosjek * 2.1*1.07;
    if (nazad == HIGH){
      smjer_radijani -= smjer;
    }
    else{//{ Izračunavanje pomaka u x i y
      smjer_radijani += smjer;
    }
    dx =  d * sin(smjer_radijani);
    dy =  d * cos(smjer_radijani);
```

```
  if (nazad == HIGH){
      x -= dx;
      y -= dy;
  }
  else{
     x += dx;
     y += dy;}
   // Provjerava u kojem kvadrantu se točka nalazi i dodjeljuje vrijednost z
 if (x >= 0 && y >= 0  ) {
  z = 1; // Kvadrant I
 } else if (x < 0 && y >= 0) {
  z = 2; // Kvadrant II
 } else if (x < 0 && y < 0) {
  z = 3; // Kvadrant III
 } else if (x >= 0 && y < 0) {
  z = 4; // Kvadrant IV
 }
 // racunanje brzine
 int brojaczaC=0;
 brojaczaC++;
 dd+= d;
 cc=dd/0.1;
 ccc+=cc;
 c= ccc/brojaczaC++;
 String message = String(x) + "," + String(y) + "," + String(z) + "," + String(c);
 SerialBT.println(message);
 //memorija NVS ovaj kodsprema svu memorije te se moze iscitat svaka prijasnja kordinata samo morja znati pod kojim je brojem countera memorija se prise tek kada upalimo ponovo kod
 static int counter = 0; // Brojač za spremanje više setova podataka
 // Kreiranje jedinstvenog ključa za svaki set koordinata
```

```
 String keyX = "x" + String(counter);
 String keyY = "y" + String(counter);
// Spremanje koordinata u NVS kao float
 preferences.putFloat(keyX.c_str(), x);
 preferences.putFloat(keyY.c_str(), y);
 counter++; // Povećanje brojača za sljedeći set podataka
 BrojPromjenaSatanja1 = 0;
BrojPromjenaSatanja2 = 0;
//Serial.println(BrojPromjenaSatanja1);
//Serial.println(BrojPromjenaSatanja2);
}
void setup() {
   pinMode(leftMotorForwardPin, OUTPUT);
   pinMode(leftMotorBackwardPin, OUTPUT);
   pinMode(rightMotorForwardPin, OUTPUT);
   pinMode(rightMotorBackwardPin, OUTPUT);
   Serial.begin(115200);
//bluethoot
    SerialBT.begin("ESP32_BT_Client");
      //memorija NVS - Inicijalizacija NVS
 preferences.begin("my-app", false);
 // Brisanje NVS memorije pri svakom pokretanju
 preferences.clear();
    // Kreiranje timera za interupt
 myTimer = xTimerCreate("MyTimer",          // Naziv timera
                        TIMER_PERIOD,       // Period timera
                        pdTRUE,             // Auto-reload, timer se resetira
                        (void *)0,          // ID timera
                        TimerCallback);     // Funkcija koja se poziva na istek timera
 // Pokretanje timera
```

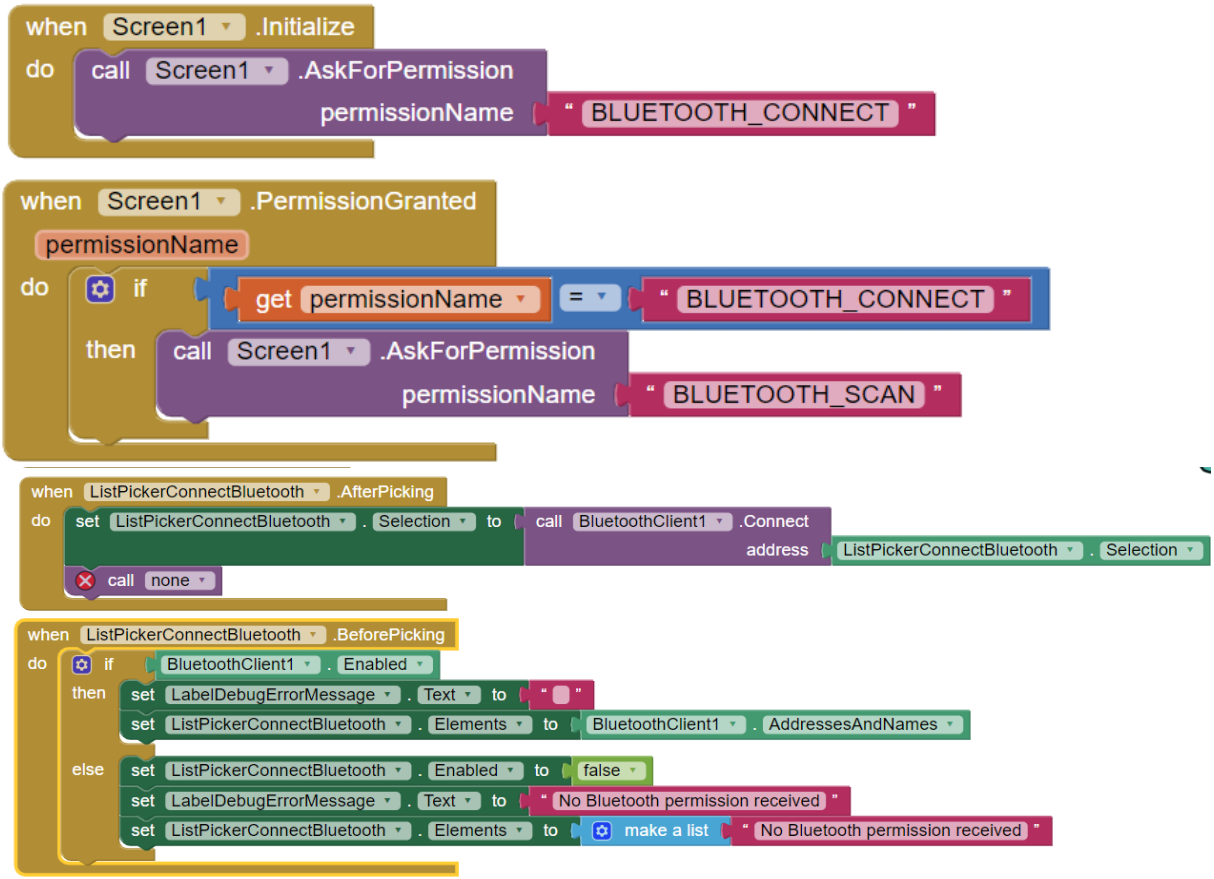
```
  xTimerStart(myTimer, 0);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}
void loop() {
if (SerialBT.connected()) {
  speed1 = 107;
  speed2 = 100;
  speed3 = 255;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  // Izračun udaljenosti
  distance = duration * SOUND_SPEED/2;
  //Serial.println(distance);
  if (distance > -1 && distance < 20 ) {
    stop();
  analogWrite(ena, 130);
  analogWrite(enb, 130);
  nazad = HIGH;
    if(random(0, 2) == 0) {
      goLeft();
    }
    else {
      goRight();
    }
    nazad = LOW;
```

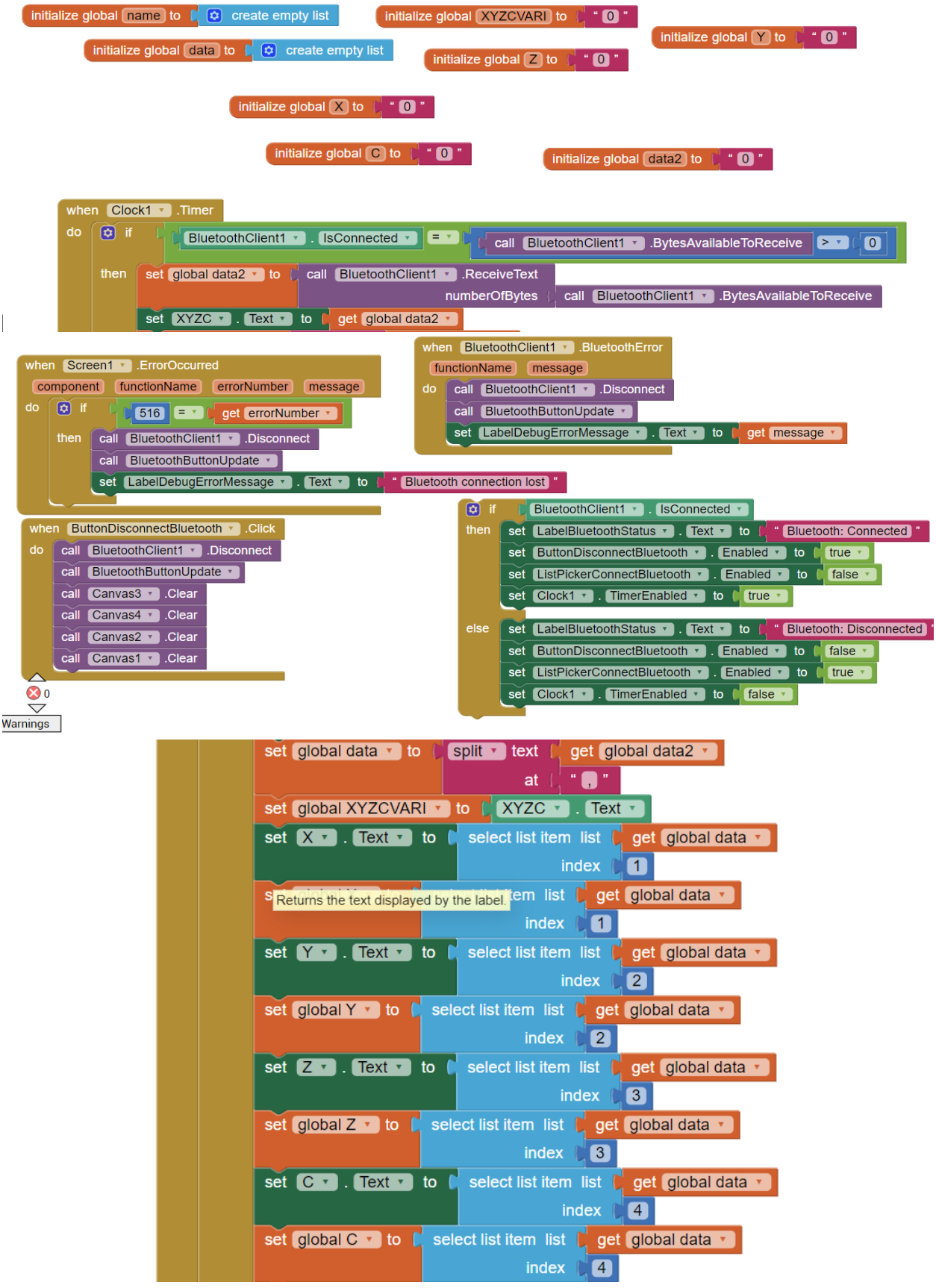
```
  }
  else {
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= interval) {
      goForward();
      previousMillis = currentMillis;
    }
    else{
      goForward2();
    }
}
}
}
void IRsensori() {
      // Čitanje stanja infracrvenog senzora (HIGH ili LOW)
  int trenutnoStanje1 = digitalRead(irPin1);
  // IR SENZORI
  // Provjera je li došlo do promjene stanja
  if (trenutnoStanje1 != prethodnoStanje1) {
    if (trenutnoStanje1 == HIGH) {
      //Serial.println("Detektiran je predmet!NA 1");  // Ispis poruke
    } else {
      //Serial.println("Nema detektiranog predmeta.NA 1");  // Ispis poruke kada predmet nije detektiran
    }
    BrojPromjenaSatanja1 ++;
    BrojPromjenaSatanja11++;
  }
  prethodnoStanje1 = trenutnoStanje1;
  // Čitanje stanja infracrvenog senzora (HIGH ili LOW)
  int trenutnoStanje2 = digitalRead(irPin2);
```

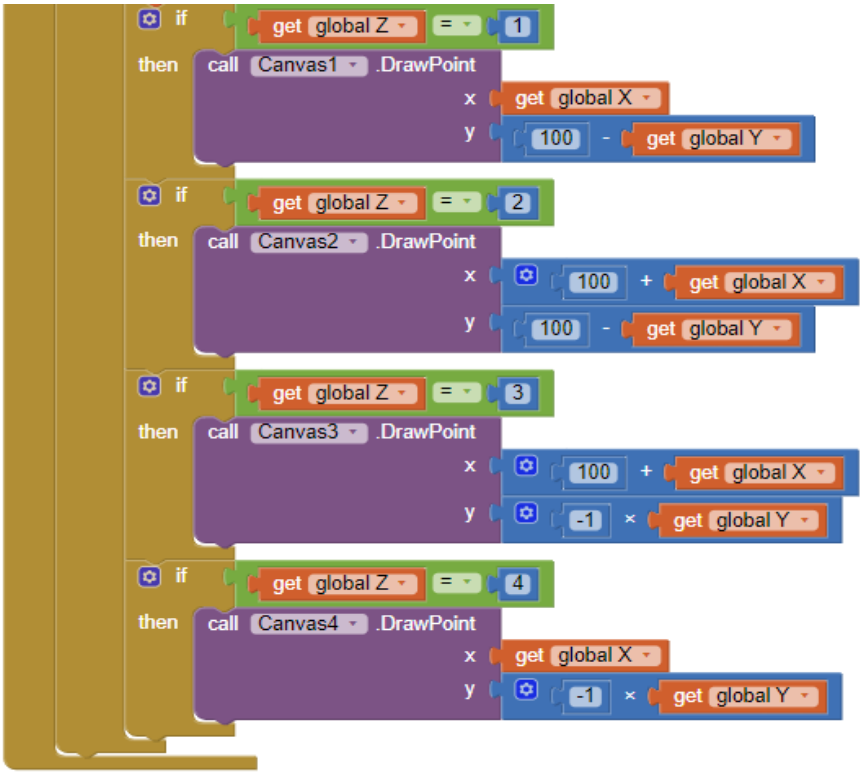
```
  // Provjera je li došlo do promjene stanja
  if (trenutnoStanje2 != prethodnoStanje2) {
    if (trenutnoStanje2 == HIGH) {
      //Serial.println("Detektiran je predmet!NA 2");  // Ispis poruke
    } else {
      //Serial.println("Nema detektiranog predmeta. NQ 20");  // Ispis poruke kada predmet nije
detektiran
    }
    BrojPromjenaSatanja2 ++;
      BrojPromjenaSatanja22++;
  }
  prethodnoStanje2 = trenutnoStanje2;
  delay(10);
}
void goForward() {
  digitalWrite(leftMotorForwardPin, HIGH);
  digitalWrite(leftMotorBackwardPin, LOW);
  digitalWrite(rightMotorForwardPin, HIGH);
  digitalWrite(rightMotorBackwardPin, LOW);
  analogWrite(ena, speed3);
  analogWrite(enb, speed3);
  IRsensori();
}
void goForward2() {
  digitalWrite(leftMotorForwardPin, HIGH);
  digitalWrite(leftMotorBackwardPin, LOW);
  digitalWrite(rightMotorForwardPin, HIGH);
  digitalWrite(rightMotorBackwardPin, LOW);
  analogWrite(ena, speed1);
  analogWrite(enb, speed2);
```

```
  IRsensori();
}
void stop() {
  unsigned long startTime = millis();
  int duration =1400;
  while (millis() - startTime < duration) {
     digitalWrite(leftMotorForwardPin, LOW);
     digitalWrite(leftMotorBackwardPin, LOW);
      digitalWrite(rightMotorForwardPin, LOW);
      digitalWrite(rightMotorBackwardPin, LOW);
      IRsensori();
  }
   Serial.println(BrojPromjenaSatanja11);
  Serial.println(BrojPromjenaSatanja22);
  delay(1000);
}
void goRight() {
  unsigned long startTime = millis();
  int duration =1400;
  while (millis() - startTime < duration) {
     digitalWrite(leftMotorForwardPin, LOW);
     digitalWrite(leftMotorBackwardPin, LOW);
      digitalWrite(rightMotorForwardPin, LOW);
      digitalWrite(rightMotorBackwardPin, HIGH);
      IRsensori();
  }
  stop();
}
void goLeft() {
  unsigned long startTime = millis();
```

```
 int duration =1400;
 while (millis() - startTime < duration) {
    digitalWrite(leftMotorForwardPin, LOW);
    digitalWrite(leftMotorBackwardPin, LOW);
     digitalWrite(rightMotorForwardPin, LOW);
     digitalWrite(rightMotorBackwardPin, HIGH);
     IRsensori();
 }
 stop();
}
```




initialize global name to create empty list
initialize global XYZCVARI to " 0 "
initialize global Y to " 0 "
initialize global data to create empty list
initialize global Z to " 0 "
initialize global X to " 0 "
initialize global C to " 0 "
initialize global data2 to " 0 "
when Clock1 .Timer
do if BluetoothClient1 . IsConnected = call BluetoothClient1 .BytesAvailableToReceive > 0
then set global data2 to call BluetoothClient1 .ReceiveText
numberOfBytes call BluetoothClient1 .BytesAvailableToReceive
set XYZC . Text to get global data2
when Screen1 .ErrorOccurred
component functionName errorNumber message
do if 516 = get errorNumber
then call BluetoothClient1 .Disconnect
call BluetoothButtonUpdate
set LabelDebugErrorMessage . Text to " Bluetooth connection lost "
when BluetoothClient1 .BluetoothError
functionName message
do call BluetoothClient1 .Disconnect
call BluetoothButtonUpdate
set LabelDebugErrorMessage . Text to get message
when ButtonDisconnectBluetooth .Click
do call BluetoothClient1 .Disconnect
call BluetoothButtonUpdate
call Canvas3 .Clear
call Canvas4 .Clear
call Canvas2 .Clear
call Canvas1 .Clear
0
Warnings
if BluetoothClient1 . IsConnected
then set LabelBluetoothStatus . Text to " Bluetooth: Connected "
set ButtonDisconnectBluetooth . Enabled to true
set ListPickerConnectBluetooth . Enabled to false
set Clock1 . TimerEnabled to true
else set LabelBluetoothStatus . Text to " Bluetooth: Disconnected "
set ButtonDisconnectBluetooth . Enabled to false
set ListPickerConnectBluetooth . Enabled to true
set Clock1 . TimerEnabled to false
set global data to split text get global data2
at " , "
set global XYZCVARI to XYZC . Text
set X . Text to select list item list get global data
index 1
Returns the text displayed by the label.
set Y . Text to select list item list get global data
index 2
set global Y to select list item list get global data
index 2
set Z . Text to select list item list get global data
index 3
set global Z to select list item list get global data
index 3
set C . Text to select list item list get global data
index 4
set global C to select list item list get global data
index 4


if
get global Z
=
1
then
call Canvas1 .DrawPoint
x
get global X
y
100
-
get global Y
if
get global Z
=
2
then
call Canvas2 .DrawPoint
x
100
+
get global X
y
100
-
get global Y
if
get global Z
=
3
then
call Canvas3 .DrawPoint
x
100
+
get global X
y
-1
×
get global Y
if
get global Z
=
4
then
call Canvas4 .DrawPoint
x
get global X
y
-1
×
get global Y