Mobilni manipulator za detekciju i uklanjanje prepreka

Maričević, Josip

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

MOBILNI MANIPULATOR ZA DETEKCIJU I UKLANJANJE PREPREKA

Završni rad

Josip Maričević

Osijek, 2019.

Sadržaj

1. UVOD 1

1.1. Zadatak i ciljevi rada 1

2. SUSTAV MOBILNG MANIPULATORA 2

2.1. Ideja strukturnog rješenja 2

2.2. Ideja upravljanjačkog algoritma 3

2.3. Podsustavi mobilnog manipulatora 4

2.3.1. Mobilnost manipulatora 4

2.3.2. Detekcija prepreka 4

2.3.3. Manipulacija i uklanjanje prepreka 4

3. REALIZACIJA MOBILNOG MANIPULATORA 5

3.1. Korišteni alati pri izradi 5

3.2. Struktura manipulatora 6

3.2.1. Mobilnost 7

3.2.2. Detekcija 7

3.2.3. Manipulacija 7

3.3. Programska realizacija 8

3.3.1. Programski kod mobilnosti 9

3.3.2. Programski kod detekcije 9

3.3.3. Programski kod manipuliranja 9

3.3.4. Iscrtavanje radara 10

4. TESTIRANJE 11

4.1. Cilj testiranja 11

4.2. Rezultati testiranja 11

4.2.1. Testiranje senzora 11

4.2.2. Kalibracija pogona 12

5. ZAKLJUČAK 14
6. LITERATURA 15
SAŽETAK 16
ABSTRACT 16
ŽIVOTOPIS 17
PRILOZI 18

Figure

Slika 2-1 Dijagram toka procesa 3
Slika 3-1 ST-Link V2 USB programator 6
Slika 3-2 Dijagram toka podataka i naredbi 8
Slika 3-3 Primjer iscrtanog radara 10
Slika 4-1 Grafički prikaz testiranja senzora udaljenosti 11
Slika 4-2 Grafički prikaz kalibracije pogona za kretanje naprijed 12
Slika 4-3 Grafički prikaz kalibracije pogona zakretanja 13
Slika 0-1 Mikrokontroler STM32F103C8T6 (Blue Pill) 20
Slika 0-2 Princip upravljanja motorom pomoću H-mosta 21
Slika 0-3 L298N modul za upravljanje motorima 21
Slika 0-4 HC-SR04 Ultrazvučni senzor 22
Slika 0-5 SG90 Servo motor 23
Slika 0-6 Robotska ruka 23
Slika 0-7 Kotač i istosmjerni motor 24
Slika 0-8 Platforma s kotačima 24

Tablica 4-1 Testiranje senzora udaljenosti 11
Tablica 4-2 Kalibracija pogona za kretanje naprijed 12
Tablica 4-3 Kalibracija pogona za zakretanje 12

Jednadžba 1 Računanje udaljenosti 22

1. UVOD

Još od davnih vremena, ljudi su radili na tome da si olakšaju život i svakodnevne radne aktivnosti. Za to su koristili razne alate koje su koristili kao pomagala. Kako je vrijeme prolazilo, tako su ti alati postajali sve složeniji i napredniji. U današnje vrijeme sasvim je normalno vidjeti stroj koji u potpunosti zamjenjuje čovijeka te samostalno manipulira sa svojom okolinom. Prvi oblik manipulacije je razvijen još 1940-ih za potrebe rada s radioaktivnim tvarima. Tada su ljudi izravno upravljali sa takvim manipulatorima, jer tehnologija nije bila dovoljno razvijen kao danas, kada imamo napredne industrijske robote koji sami određuju svoju unaprijed definiranu funkciju. Takve strojeve nazivamo autonomni manipulatori. Autonomna manipulacija danas predstavlja izazov u svim robotskim tehnologijama. Odnosi se na sposobnost mobilnog robotskog sustava s jednim ili više manipulatora koji obavlja interventne zadatke koji zahtijevaju fizičke kontakte u nestrukturiranim okruženjima i bez kontinuiranog ljudskog nadzora. Postizanje sposobnosti autonomne manipulacije je kvantni skok u robotskim tehnologijama jer je trenutno izvan najsuvremenije robotike. Ukoliko ovakvom stroju dodamo još i mogućnost kretanja, dobijemo mogućnost manipulacije i mobilnosi. U ovom slučaju potrebna je koordinacija robotske ruke i platforme. To je specifičnost mobilne manipulacije. U većini zadataka korisnici moraju kontrolirati mjesto, položaj i orijentaciju robotske ruke te položaj platforme, dok će se u ovome radu pokušati realizirati oblik autonomne mobilne manipulacije. Ovakvi uređaji se koriste u rafinerijama, petrokemisjkim i kemijski industrijama gdje je potrebno prikupiti uzorke iz kontaminiranih okolina. Uređaj zamjenjuje čovjeka te sprječava njegovo izlaganje. Jedan primjer wsu i mali autonomni usisavači koji samostalno prepoznaju svoju okolinu, te prema tome određuju koji je korak potrebno poduzeti idući.

1.1. Zadatak i ciljevi rada

Ovaj rad za zadatak ima dizajnirati, realizirati te testirati autonomni mobilni manipulator, koji bi se koristio za prepoznavanje i uklanjanje prepreka sa puta. Cilj je na platformu dograditi sve potrebne komponente kako bi manipulator mogao odrediti položaj prepreke, dovesti se do njega, te ga robotskom rukom podići i skloniti s puta. Za pronalaženje prepreke, koristi će se primjer iz okoline, točnije ultrazvučni valovi. Pomoću njih će se odrediti položaj i udaljenost od predmeta, kako bi se manipulator moga približiti predmetu, te ga ukloniti sa puta. Za vrijeme skeniranja, pomoću bluetooth komunikacije, vrijednosti će se slati na računalo i iscrtavati radar.

2. SUSTAV MOBILNG MANIPULATORA

Kako je već navedeno gore, ljudi se sve češće zamjenjuju sa strojevima. Razlozi za to su razni. Neki od njih su: olakšanje ljudima, zamjena ljudi, jeftinija proizvodnja, veća preciznost, kvliteta i brzina izrade. Isto tako ih možemo korisiti i u situacijama u kojima je ugrožena ljudska sigurnost. Ovaj rad je zamišljen tako da manipulator uklanja prepreke sa puta, što može biti jako korisno ako su u pitanju neki opasni predmeti. U tome slučajnu se predmet može ukloniti, bez ljudske interakcije s njime. Problem ovoga zadatka je kako pomoću senzora za udaljenost prepoznati predmet na ravnoj podlozi, te sa istim tim senzorom navigirati manipulator do toga predmeta i pozicionirati ga tako da ga isti pokupi sa svojom robotskom rukom. Potrebno je programirati tu ruku tako da se ona ispruži i pokupi taj predmet, te ga zatim odloži pored manipulatora. Manimuplator će se sastojati od platforme, robotske ruke, kotača i ultrazvučnog senzora.

2.1. Ideja strukturnog rješenja

Pri izradi ovoga manipulatora, bit će potrebno tako ga dizajnirati da na jednu platformu možemo smjestiti sve potrebne komponente, na način da one mogu obavljati svoju funkciju i zadaću neometano. Platforma će biti plastična podloga na koju je potrebno smjestiti ostale komponente, kako je već navedeno prije. Određivanje položaja predmeta ce se određivati ultrazvučnim senzorom udaljenosti, te ga je potrebno smjestiti s donje prednje strane platforme kako bi imao dobar pregled na prostor ispred sebe. Tri kotača od kojih će dva biti pogonska potrebno je postaviti tako da platforma bude stabilna. Manipulator mora imati mogućnost kretanja naprijed i nazad, te se isto tako i motori koji ih pokreću moraju kretati u oba smjera. Za realizaciju ovoga će se koristi jednostavan modul za upravljanje istosmjernih motora.

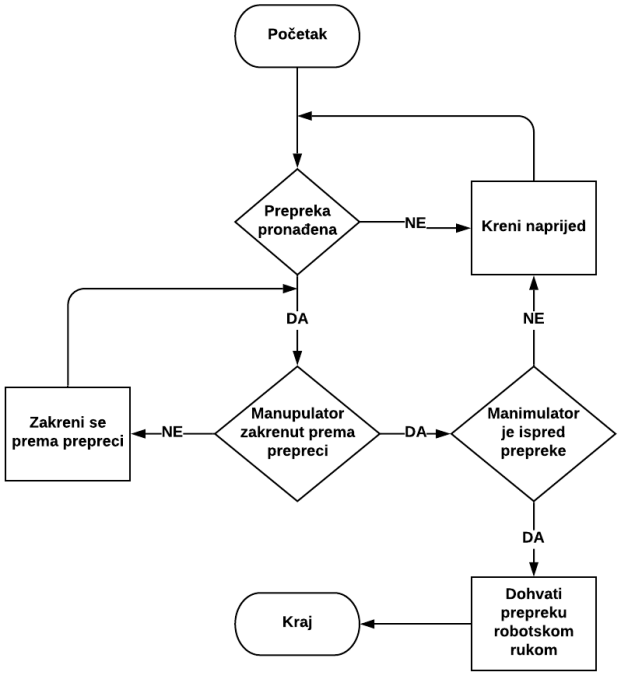
Robotska ruka koja je potrebna za podizanje predmeta mora biti realizirana tako da može dosegnuti do površine na kojoj manipulator operira, kako bi mogla uhvatiti predmet i podići ga. Ona mora imati nekoliko pokretnih dijelova koji će omogućiti kretanje u više smjerova, i nekakvim kliještima kojima će se prihvatiti predmet.

Svi ovi dijelovi će biti upravljani pomoću STM32F103 mikrokontrolera. O samom upravljanu će nešto više biti napisao u idućem potpoglavlju.

Sve komponente manipulatora biti će napajane sa zajedničke baterije, koja će po potrebi biti spojena pomoću naponskog pretvrača.

2.2. Ideja upravljanjačkog algoritma

Kako je već navedeno, za upravljanje cijeloga manipulatora će biti zadužen STM32 mikrokontroler. Za upravljanje će biti potrebno omogućiti pinove za upravljanje motora, i primanje informacija sa senzora. Za upravljanje pogonskih motora potrebno je samo na modul za upravljanje motora poslati signal s visokom vrijednosti, na pin koji odgovara potrebnom smjeru odgovarajućeg motora. Svi ostali motori kojima će se upravljati su servo motori, a za njihovo upravljanje će se realizirati pomoću pulsno širinske modulacije signala. Da bi dobili pulsno širinsku modulaciju signala, potrebno je omogućiti korištenje brojača (eng. *Timer*) i pinova koji omogućuju korištenje tih brojača. Sam upravljački logaritam je osmišljen tako da se upravljanje odvija u sekvencama koje će se redom izvoditi. Svaka sekvenca će se izvoditi dok se ne postigne ono što je zadano, te će zatim pokrenuti iduću. Svaka sekvenca će biti smjestena u u svoju funkciju unutar samog programa. Rad samog manipulatora pokrenuo bi korisnik, i time započeo sa rednim izvođenjem sekvenci. Nekada će se sekvence izvoditi više puta, da bi postigli bolje rezultate, preciznije kretanje i manipuliranje. Sva će se mjerenja pomoću bluetooth modula slati na računalo i iscrtavati radar.



Slika 2-1 Dijagram toka procesa

2.3. Podsustavi mobilnog manipulatora

2.3.1. Mobilnost manipulatora

Da bi manipulator bio mobilan, potrebno mu je omogućiti nekakvu vrstu kretanja, odnosno mobilnosti. Pokretljivost manipulatora će nam omogućiti kotači sa pripadnim istosmjernim motorima. Na zadnjoj strani platform, svake strane će biti montiran jedan pogonski kotač, a na prednjoj strani te će biti jedan manji kotač sa mogućnsti zakretanja. Motori pogonskih kotača će biti spojeni na bateriju, a kako bi omogućili pokretanje u oba smjera, bit će potrebno motore spojiti na H-most prije spajanja na bateriju. On će biti upravljan od strane glavnom mikrokontrolera. Manipulator će imati mogućnost kretanja naprijed, natrag, te zakretanje u obje strane. Zakretanje će se odvjati na mjestu jer će mu to moći omogućeno sa prednjim zakretnim kotačićem. Nakon što manipulator detektira prepreku, morat će se približiti njoj.

2.3.2. Detekcija prepreka

Ovaj mobilni manipulator će se koristiti za uklanjanje prepreka sa puta, koje prethodno mora sam detektirati. To će mu omogućiti senzor za mjerenje udaljenosti. Bit će smješten na prednjem dijelu platforme kako bi neometano moga pretraživati prostor ispred manipulatora. Kako bi senzor imao mogućnost detekcije u više smjerova, omogućit ćemo mu zakretanje u oba smjera. To ćemo izvesti tako da ga na platformu montiramo pomoću servo motora. Detekcija će se izvoditi na principu radara. Senzor ćemo zakretati i na svakom koraku mjeriti udaljenost. Daljnim usproređivanjem mjerenja ćemo lako zaključiti dali postoji prepreka ispred manipulatora, gdje je ona i na kojoj udaljenosti.

2.3.3. Manipulacija i uklanjanje prepreka

Kako smo gore rekli da je manipulator zadužen za detekciju i uklanjanje prepreka, potrebno mu je omogućiti način na koji će ukloniti prepreku nakon njezine detekcije. Kada prepreka bude u dobrom položaju za otklanjanje, odnosno kada se manipulator približi prepreci na određenu udaljenost, manipulator započinje sa otklanjanjem prepreke. Uklanjanje prepreke i manipulacija će se izvoditi pomoću robotske ruke, izvedene na jednostavan način pomoću servo motora. Ruka će također biti upravljana od strane mikrokontrolera. Zadaća robotske ruke je da dohvati predmet ispred manipulatora i spusti je na sam manipulator.

3. REALIZACIJA MOBILNOG MANIPULATORA

Izrada ovoga zadatka je zamišljena pomoću 'STM32F1' ARM mikrokontrolera koji bi upravljalo 'HC-SR04' ultrazvučnim senzorom za mjerenje udaljenosti, te prko 'L298N' modula upravljao sa dva motora koja se koriste za pokretanje. Za zakretanje senzora i pokretanje robotske ruke koristiti će se mali servo motori 'SG90'.

Manipulator se pokreće pritiskom korisnika na tipku za pokretanje. Nakon pokretanaj potrebno je senzorom pretražiti pritor ispred sebe i pronaći predmet. Pretraga predmeta izgleda poput radarskog skeniranja. Ultrazvučni senzor se zakreće u jednu stranu, te se u sitnim koracima zakreće do druge strane. Pri svakom koraku program koji je upisan u mikrokontroler sprema vrijednosti koje mu daje senzor. Ako je predmet dovoljno blizu senzoru, prilikom okretanja senzora prema predmetu, primjeti se veliko odskakanje u mjerenju vrijednosti. Iz dobivenih podataka, proračuna se u kojem smjeru odnosto na manipulator se nalazi predmetu. Prethodnim mjerenjima i testovima, izračunato je kako upravljati sa motorima da bi ga okrenuli za točno određeni kut. Kako sada imamo smjer, odnosno kut na kojemu se predmet nalazi, lako možemo manipulator zakrenuti točno prema sredini predmeta. Zatim će senzor mjeriti udaljenost od predmeta dok mu se manipulator ne približi na točnu određenu udaljenost. Po poterebi se može zaustaviti približavanje, te još jednim skeniranjem provjeriti dali se manipulator kreće u pravom smjeru. Manipulator treba točno postaviti ispred predmeta da bi ga ruka moga dohvatiti. S obzirom da koristimo samo jedan senzo, predmet treba stajati na točno određenom mjestu za koje je ruka programiran da dohvaća redmet. Robotska ruka je zadužena da se nakon prilaska predmetu, zakrene i približi predmetu otvorenih hvataljki, te pomoću njih uhvati predmet. Predmet zatim podiže i zakreće iznad sebe, kako nebi iskliznuto natrag na put.

3.1. Korišteni alati pri izradi

Pri izradi ovoga rada koristio sam mnoge razvojne alate. Prije samog početka programiranja mikrokontrolera pomoću programa STM32CubeMX bilo je potrebno generirati incijalni kod. STM32CubeMX je grafički alat izdan od strane STMicroelectronics-a za konfiguraciju i inucijalizaciju za *Arm Cortex* mikrokontrolere. On nam nakod odabira razvojne pločice, odnosno mikrokontrolera, omogiću konfiguraciju istog konrolera. Konfigurirati možemo *clock* mikrokontrolera, unutrašnje tajmere, te mu odrediti sve ulazne i izlazne GPIO (General Purpose Input Output) portove. Nakon konfiguriranja, ovj alat će nam generirati inicijalizirati početni kod koji pokreće sve unutršnje sustave potrebe za rad mikrokontrolera i svih njegovih portova. Cijeli generirani projekt zatim otvorimo sa drugim alatom, MDK 5 Keil. Keil je alat za razvoj softvrskih rješenja (*eng. Microcontroller Development Kit – MDK)* ARM mikrokontrolera.

On nam omogućuje da uredimo programski kod i dodamu svu potrebnu logiku koja povezuje ulaze i izlaze is kontrolera. Također pomoću njega cijeli projektni kod zapišemo u kontroler. Naravno prije zapisivanja potrebno je povezati mikrokontroler s računalom, a to sam realizirao pomoću ST-LINK/V2 programatora i debuggera, koji je također izdan od strane STMicroelectronics-a.

Slika 3-1 ST-Link V2 USB programator

ST-LINK Utility je potpuno funkcionalno softversko sučelje za programiranje STM32 mikrokontrolera. Pruža jednostavno i učinkovito okruženje za čitanje, pisanje i provjeru memorijskog uređaja. Za krajnje programiranje mikrokontrolera, koristilo se Arduino IDE[1] razvojno okruženje. Pošto se Arduino IDE koristi samo za Arduino razvojne pločice, potrebno je dodati repozitorij STM32duino, s STM32F103 kontrolerom koji će se koristiti u izradi.

3.2. Struktura manipulatora

Manipulator je izveden tako, da je napravljena čvrsta platforma od prozirne plastike, na koju su pričvršćena 3 kotača. Jedan manji se nalazi na prednjem djelu platforme i može se zakretati u svim smjerovima. Dok su zadnja dva pogonska kotača puno veća, i montirana su svaki na svoj motor. Oba motora su pomoću plastičnih okvira pričvršćena za plastičnu platformu. Senzor za mjerenje udaljenosti je pričvršćen za servo motor koji je smješten na prednji dio platforme. Robotska ruka je sastavljena od 3D printanih dijelova u koje su umetnuti servo motori. Ruka je smještena na platformu tako da se može zakretati. Za napajanje servo motora, pogonskih motora te ostalih kontrolera se koristi baterija koja je pričvršćena na zadnjem kraju platforme. Na bateriju se prvo spaja Step_Down[2] pretvarač koji će napjanje baterije spustiti na 5V. Pretvarač, modul za

[1] IDE – Integrirano razvojno okruženje (Integrated Development Environment) je softverska aplikacija koja računalnim programerima pruža sveobuhvatne pogodnosti
[2] Step-Down pretvarač – pretvarač koji smanjuje vrijednost napona, ne mjanjajući pri tome vrijednost frekvrncije i oblik napona

upravljanje motorima i mikrokontroler su smješteni na donju zadnju stranu platforme kako nebi zauzimale mesto za ostale pokretne djelove. Bluetooth modul koji je potreban za iscrtavanje radara je također spojen na napajanje i povezan s mikrokontrolerom.

3.2.1. Mobilnost

Kako bi manipulator mogao dohvatiti prepreku, potrebno je omogućiti mu neku vrstu kretanja, kako bi se on prethodno mogao primaći to prepreci. Kretanje mnipulatora je realizirano na način da su mu s donje strane platforme postavljena tri kotača. Jedan je na prednjoj strani i može se slobodno rotirati u svim smjerovima. Dva veća pogonska motora su montirana s stražnje strane manipulatora koja omogućuju kretanje prema naprijed, natrag te zakretanje u obje strane. Svaki od motora je spojen na svoj reduktor, na koji je prčvršćen kotač. Reduktori su nam potrebni jer je brzina motora prevelika, a smanjenjem brzine dobijamo jači potezni moment koji je potreban za zakretanje kotača. Motori se upravljaju s H-mostom za upravljanje DC[3] motora. On nam omogućuje zakretanje motora u obje strane na jednostavan način, uz mogućnost određivanja snage za svaki od motora zasebno. Za napajanje H-mosta koriste se 3 6LR61 baterije. Isti napon se koristi za upravljanje motora, koji je pomoću PWM-a[4] smanjen na 80% snage. H-mostom se upravlja pomoću 6 pinova, 2 pina za upravljanje PWM-om za svaki motor, a 4 za upravljanje motorima. Motor se pokreće tako da dovedemo visoku vrijednost napona na određeni pin koji pokreće jedan motor u određenom smjeru. Motorima je moguće upravljati u isto vrijeme u svim smjerovima.

3.2.2. Detekcija

Da bi se moglo upravljati motorima, potrebno je prvo znati u kojem smjeru se manipulator treba kretati ili zkretati. Položaj prepreke se određuje pomoću ultrazvučnog senzora. Senzor je montiran na servo motor. Motor se zakreće od lijeve strane prema desnoj po jedan stupanj, te za svaki pomak senzor mjeri udaljenost. Kada dođe do kraja, isti postupak se izvodi samo od desne strane prema lijevoj. Na osnovu svih mjerenja određuje se udaljenost i kut pod kojem se nalazi prepreka, u slučaju da je ista pronađena. Ti podatci se predaju sustavu za mobilnost, koji u skladu s njima pomiče manipulator.

3.2.3. Manipulacija

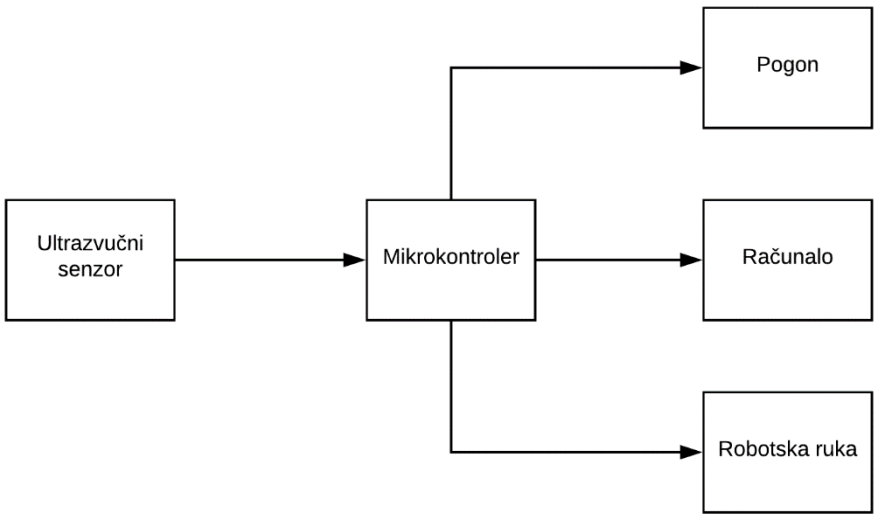
Nakon detekcije i primicanja manipulatora prepreci, potrebno je tu prepreku skloniti sa puta. Za to je odgovorna robotska ruka koja se sastavljena od 6 servo motora i 3D printane plastične konstrukcije. Pomoću senzora manipulator se dovede točno 3cm daleko od prepreke, te tek tada robotska ruka može započeti uklanjanje prepreke. Ruka kreće iz desnog položaja prema naprijed,

[3] DC – Istosmjerna struja (Direct Current)
[4] PWM – Pulsno širinska modulacija (Pulse Width Modulation) je metoda smmanjenja prosječne vrijednosti napona, sjeckanjem na manje dijelove

otvara kliješta i spušta se dolje prema prepreci. Nakon toga zatvara kliješta, te ju podiže i ponovno vrća na desnu stranu, gdje ga odlaže i ostaje u tom položaju dok se ponovno ne detektira iduća prepreka. Svi servo motori će isto kao i mikrokontroler biti napajani pomoću gore navedenog StepDown pretvarača koji je priključen na baterije od 9V;

3.3. Programska realizacija

Programski kod prenesen na mikrokontroler je pisan u Arduino IDE-u pomoću C programskog jezika. Kod je realiziran tako da je podjeljen u funkcije. Na samom početku koda, prvo su definirani svi pinovi mikrokontrolera kojiće se koristiti, isto tako su definirani servo motori. Prva funkcija koja se pokreće je 'setup' i u njoj se definiraju svi pinovi koji se koriste te se priključuju svi motori i postavljaju na svoju početnu poziciju. U istoj funkciji se još i započinje serijska komunikacija s računalom. U 'loop' funkciji se pozivanjem svih preostalih funkcija skenira prostor ispred manipulatora, te zatim zakreće i primiče manipulator prema prepreci ovisno o rezultatima skeniranja. U slučaju da prepreka nije pronađena, manipulator se kreće prema naprijed te ponovno započinje skeniranje. Nakon što se približi na dovoljnu udaljenost za hvatanje robotskom rukom, okreće se algoritam uklanjanj prepreke, te se cijeli postupak ponovno pokreće.



Slika 3-2 Dijagram toka podataka i naredbi

3.3.1. Programski algoritam mobilnosti

Za pokretanje manipulatora potrebno je upravljati s H-mostom, jer je on taj koji pokreće pogonske DC motore. Za upravljanje je definirano 6 izlaznih pinova (EN_A, IN1, IN2, IN3, IN4, EN_B). Na EN_A i EN_B pinove šljemo PWM signal pomoću analogWrite() naredbe, te tako određujemo snagu motora. IN pinovi se za kontrolu kretanja. IN1 i IN2 je par pinova koji upravlja sa prvim motorom, u našem slučaju je to lijevi motor, i to tako da postavima jedan od njih na visoku razinu, a drugi na nisku razinu. Time će se naš motor kretatu u jednu stranu, a zamjenom visoke i niske razine motor će priomjeniti smjer. Ako oba pina postavimo na nisku razinu, on će se zaustaviti. Na isti način se koriste preostala dva pina za upravljanje drugim motorom. U programu su napisane funkcije za kretanje u naprijed, i zakretanje, postavljanjem ulaznih pinova na određene vrijednosti. Motori nemaju točno definiranu vrijednost zakretanja i kretanja prema naprijed, već se kreću ovisno o udaljenosti i položaju prepreke. Što znači da ako je prepreka više udaljena, oni će se više pomaći.

3.3.2. Programski algoritam detekcije

Sama detekcija prepreke je razmještena u više funkcija. Najniža je funkcija u kojoj se sammo detektira udaljenost koju senzor trenutno vidi. To je izvedeno tako da senzoru šaljemo visoku razinu na 'trig' pin u trajanju 10us, te čekamo povratnu informaciju na pinu 'echo' u obliku vremena koje kasnije preračunavamo u udaljenost.

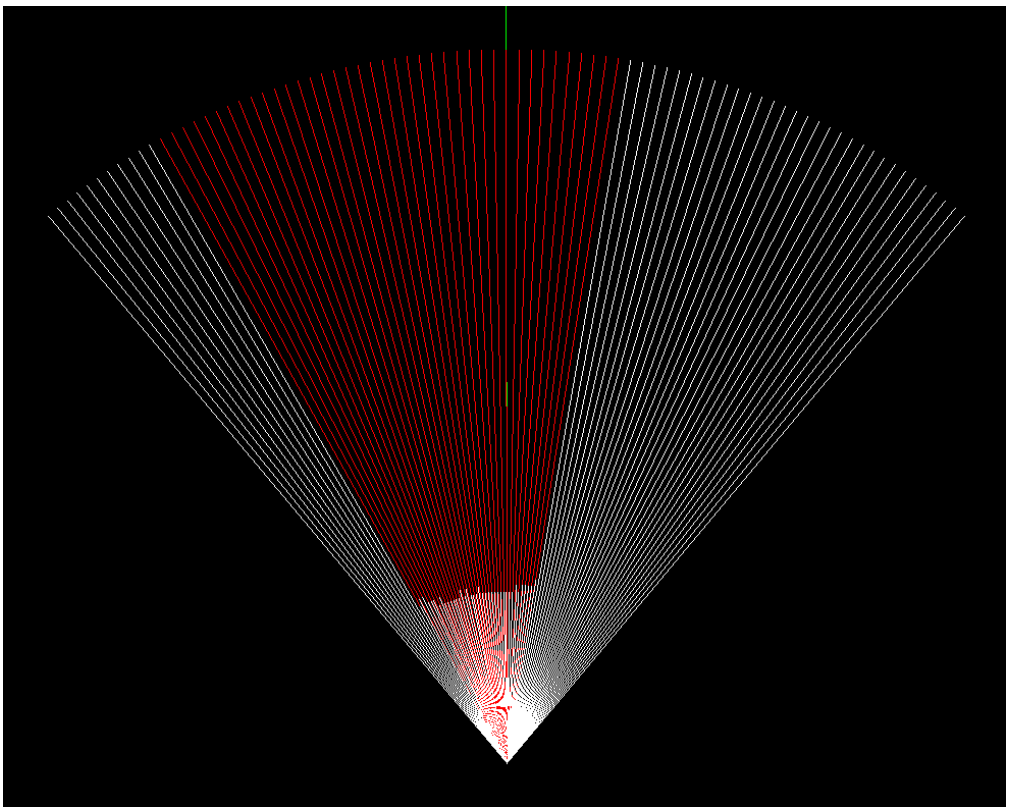
Kako je skeniranje potrebno obaviti u više smjerova, senzor se zakreće, te se za svaki pomak ponovno pokreće funkcija za mjerenje udaljenosti. Tako dobijemo skup mjerenja povezanih s pozicijom iz kojih se proračunava na kojoj je udaljenosti i kutu prepreka. Ti se podatci predaju funkcijama za mobilnost te se u odnosu na njih kreće manipulator. Ako se iz skeniranja zaključi da je prepreka na udaljenosti jednakoj i manjoj od 3cm, pokreće se algoritam za manipuliranje.

3.3.3. Programski algoritam manipuliranja

Kako smo već naveli da se u 'setup' funkciji priključuju svi servomotori i zakreću na početnu poziciju. Za pokretanje pojedinih servo motora je napravljena funkcija moveServo() kojoj se predaje pozicija i servo koji želimo pomaći, te ona samo očitava zadnju poziciju serva i pokreće servo. Pokreće ga na način da u for petlji povećava kut motora za jedan stupanj, a nakon svakog pomaka postavlja pauzu od nekoliko desetaka milisekundi, kako bi motori imali lagane i glatke pokrete, bez puno trzanja i naglih pokreta. Sva kretanja robotske ruke su smještena u novoj funkciji koja u sebi poziva prethodno navedenu funkciju za pomicanje servo motora.

3.3.4. Iscrtavanje radara

Za iscrtavanje radara potrebno je povezati mikrokontroler sa računalom, a to je realizirano pomoću bluetooth modula. Pri korištenju bluetooth modula potrebno je započeti serijsku komunikaciju s računalom, te mu slati podatke na temelju kojih će on iscrtavati radar. Pri svakom koraku prethodno objašnjene detekcije prepreke, podatci udaljenosti i trenutnog položaja se šalju račnalu. Na računalu se pomoću kratke python skripte iscrtava radar. Ona u beskonačnoj petlji iscrtava liniju za svako očitanje koje primi od mikrokontrolera. Nakon što servo s senzorom dođe do kraja, zaslon se briše i iscrtavanje kreće ponovno.



Slika 3-3 Primjer iscrtanog radara

4. TESTIRANJE

Kod većine projekata, nije dovoljno samo samo sastaviti sve dijelove i pustiti u pogon, već je potrebno prvo provesti potrebna testiranja, baždarenja i kalibriranja sustava.

4.1. Cilj testiranja

Testiranja će se provoditi na sustavu detekciju i na pokretanju manipulatora. Testirati će se točnost i preciznost mjerenja udaljenosti ultrazvučnog senzora. Drugo testiranje koje će se provoditi je zapravo kalibriranje pogona manipulatora. Pošto se za pokretanje koriste obični DC motori, potrebno je odrediti koliko dugo će oni raditi da bi se manipulator zakrenuo za određeni kut, ili pomaknuo prema naprijed.

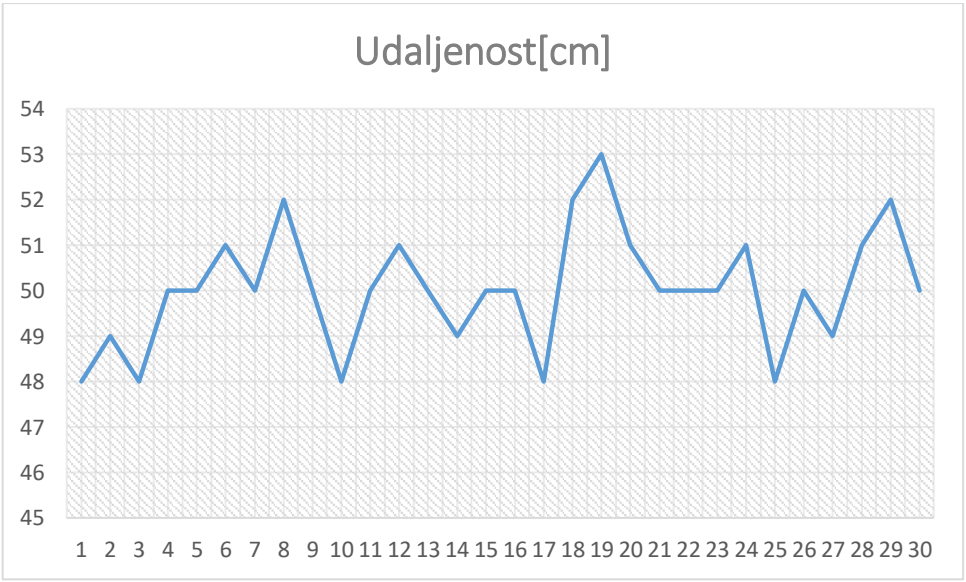
4.2. Rezultati testiranja

4.2.1. Testiranje senzora

Pri testiranju senzora, postavio sam prepreku na udaljenost točno 50cm od senzora, i mjerio udaljenost senzorom trideset puta.

Mjerenje	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Udaljenost[cm]	48	49	48	50	50	51	50	52	50	48	50	51	50	49	50
Mjerenje	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Udaljenost[cm]	50	48	52	53	51	50	50	50	51	48	50	49	51	52	50

Tablica 4-1 Testiranje senzora udaljenosti



Slika 4-1 Grafički prikaz testiranja senzora udaljenosti

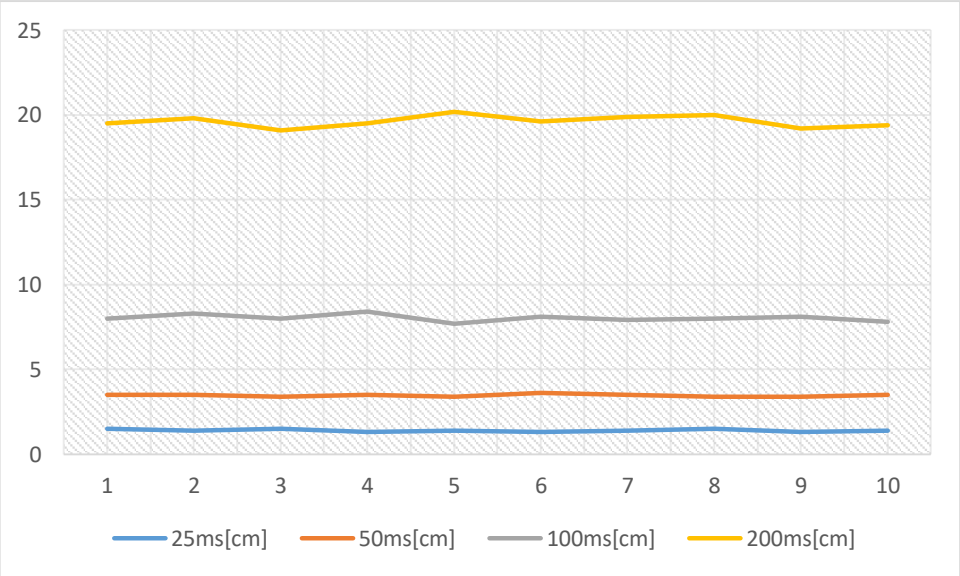
Iz prikazanih mjerenja možemo vidjeti kako senzor ne mjeri uvijek jednako, ali rezultati su približno točni.

4.2.2. Kalibracija pogona

Potrebno je kroz testiranja, izmjeriti koliko nam se manipulator zakreće ili pomiče za određeno trajanje toga kretanja. Testirano je na način da se pogonski motori puste u rad na određeno trajanje, i mjerimo pomak koji je manipulator napravio. Prvo se mjeri koliko se manipulator pomakne prema naprijed u centimetrima, ako se oba motora vrte prema naprijed 25, 50, 100 i 200 ms.

Mjerenje	1	2	3	4	5	6	7	8	9	10
25ms[cm]	1.5	1.4	1.5	1.3	1.4	1.3	1.4	1.5	1.3	1.4
50ms[cm]	3.5	3.5	3.4	3.5	3.4	3.6	3.5	3.4	3.4	3.5
100ms[cm]	8.0	8.3	8.0	8.4	7.7	8.1	7.9	8.0	8.1	7.8
200ms[cm]	19.5	19.8	19.1	19.5	20.2	19.6	19.9	20.0	19.2	19.4

Tablica 4-2 Kalibracija pogona za kretanje naprijed

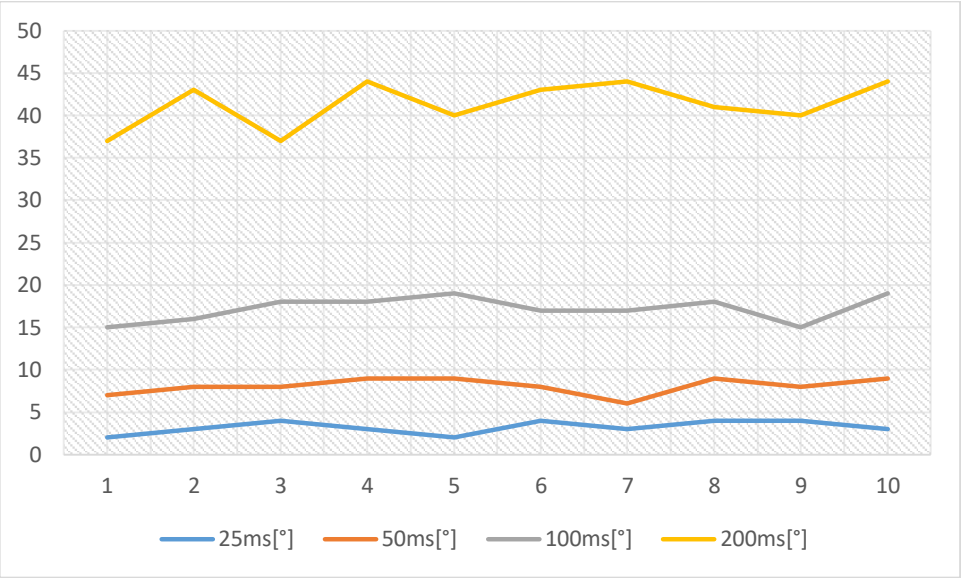


Slika 4-2 Grafički prikaz kalibracije pogona za kretanje naprijed

Na jednak način se mjeri koliko se manipulator zakrene u stupnjevima.

Mjerenje	1	2	3	4	5	6	7	8	9	10
25ms[°]	2	3	4	3	2	4	3	4	4	3
50ms[°]	7	8	8	9	9	8	6	9	8	9
100ms[°]	15	16	18	18	19	17	17	18	15	19
200ms[°]	37	43	37	44	40	43	44	41	40	44

Tablica 4-3 Kalibracija pogona za zakretanje



Slika 4-3 Grafički prikaz kalibracije pogona zakretanja

5. ZAKLJUČAK

Cilj ovoga završnog rada je bio dizajniriti, konstruirati, programirati i testirati autonomni mobilni manipulator, s funkcijom uklanjanja prepreka sa puta. Njegovim izvršavanjem prošao sam mnoga područja elektronike i fizike. Izrada je bila jako zahtjevna, ali uz mnogo truda i napora sam uspjeo izraditi ono što se zahtjevalo od mene. Jako pun sam naučio jer je bilo potrebno dobro proučiti svaku komponentu od koje je sastavljen, i svaki alat koji je korišten pri izradi. Prošao sam kroz sve sustave STM32 okoline, od mikrokontrolera do aplikacija za njihovo upravljanje i programiranje. Mislim kako će mi ova znanja jako pomoću u daljnjem studiranju, i budućem zapošljavanju.

6. LITERATURA

[1] STMiroelectronics, STM32F103C6, https://www.st.com/content/st_com/en.html, pristupljeno 25.06.2019

[2] SparkFun, L298n H-Bridge, https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf, pristupljeno 25.06.2019

[3] SparkFun, HC-SR04, https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf, pristupljeno 27.06.2019

[4] Pcheung, SG90, http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf, pristupljeno 23.06.2019

[5] G. Brown, Discovering the STM32 Microcontroller, Indiana University, 2016

[6] C. Noviello, Mastering STM32

SAŽETAK

Cilj ovog praktičnog rada bio je realizirati mobilni manipulator za uočavanje i uklanjanje prepreka na putu. Pri izradi je bilo potrebno koristiti STM32F1 mikrokontroler, ultrazvučni senzor udaljenosti te robotsku ruku.
Manipulator je morao izvesti slijedeće:

- Skenirati prostor ispred sebei pronaći prepreku
- Zakrenuti cijeli manipulator prema predmetu
- Krenuti prema prepreci i zaustaviti se ispred njega
- S robotsko rukom podići predmet i odložiti ga na sebe

Glavni problem pri izradi ovoga manipulatora su prepoznavanje prepreke pomoću ultrazvučnog senzora za udaljenost.

Pr izradi ovoga projekta koristiti ćemo motedu koja se danas koristi svagdje oko nas, a to je skeniranje pomoću ultrazvučnih valova. Neke životinje, npr. šišmiši, kitovi ili dupini, koriste ultrazvučne valove za kretanje i prepoznavanje okoline. S napretkom tehnologije, i ljudi su počeli koristiti ovaj način mjerenja. Koristi se pri određivanju udaljenosti, računanju protoka, mjerenju razine fluida ili krutina u silosima, te u medicini u obliku ultrazvuka.

ABSTRACT

The aim of this practical work was to realize a mobile manipulator for detecting and removing obstacles on the road. During the construction, it was necessary to use STM32F1 microcontroller, ultrasonic distance sensor and robotic arm. The manipulator had to do the following:

- Scan the space in front of you to find an obstacle
- Rotate the entire manipulator and face it to the object
- Drive to to the obstacle and stop in front of it
- Raise the object with the robotic arm and put it on yourself

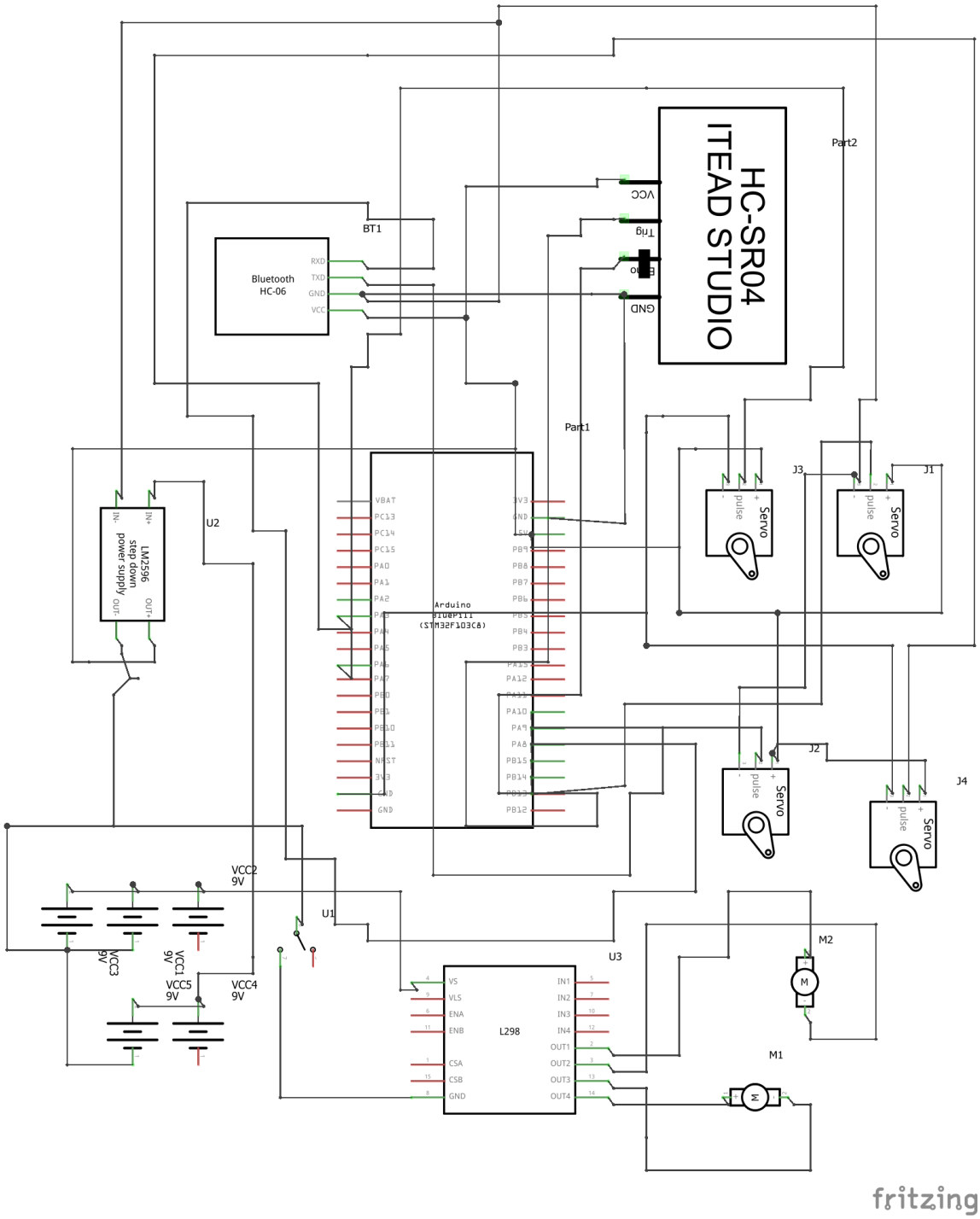
The main problem in making of this manipulator is identifying an obstacle using an ultrasonic distance sensors. For this project, we will use the method used today everywhere around us, which is scanning environment using ultrasonic waves. Some animals, e.g. bats, whales or dolphins, use ultrasonic waves to move and recognize the environment. With the advancement of technology, people began to use this method for measurements. It is used in determining distance, flow calculation, fluid level in tanks or solid level measurement in silos, and in medicine in the form of ultrasound.

ŽIVOTOPIS

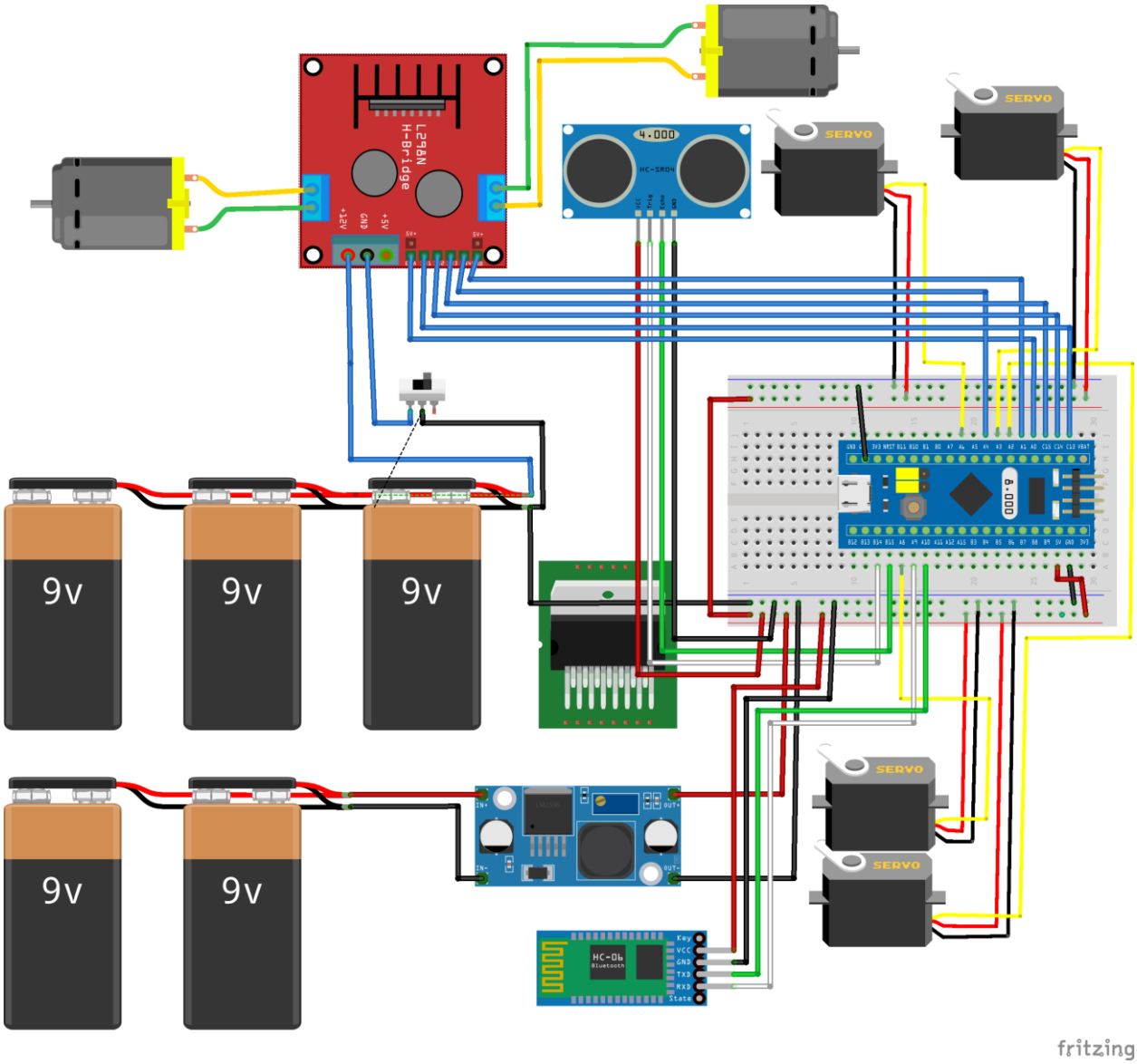
Josip Maričević dolazi iz mjesta Bodovaljci, u blizini Nove Gradiške. Završio je stručni studij automatike na Fakultetu Elektrotehnike, Računarstva i Informacijskih Tehnologija sveučilišta u Osijeku. Ima završenu srednju elektrotehničku školu u Novoj Gradiški. Samostalno grovori, čita i piše jedan stani jezik, engleski. Odradio je dvije stručne prakse, jednu u RTV servisu za računala i računalnu opremu u Novoj Gradiški, te drugu u Atos CVC-u u Osijeku.

PRILOZI

Prilog 1. Električna shema



Prilog 2. Montažna shema



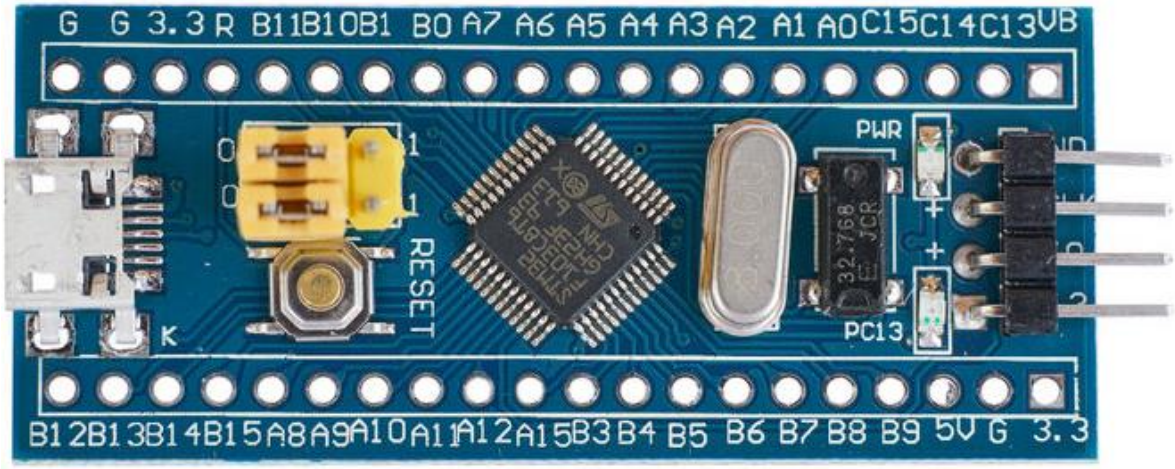
Prilog 3. Komponente projekta

Pri izradi ovoga projektnog zadatka korišteno je:

- STM32F103C8T6 Arm mikrokontroler
- L298N modula za upravljanje motorima
- HC-SR04 ultrazvučni senzor
- SG-90 servo motori
- Robotska ruka
- Istosmjerni motori s kotačima
- Plastična platforma

STM32F103C8T6 Arm mikrokontroler

STM32F103C8T6 je mikrokontroler proizveden je od strane francusko-talijanskog proizvođača *STMicroelectronics.* Pokreće ga Cortex-M3 proceser sa maksimalnom frekvencijom od 72 MHz. Sklopljen je sa 64 KB brze stalne memorije i 20KB radne memorije.

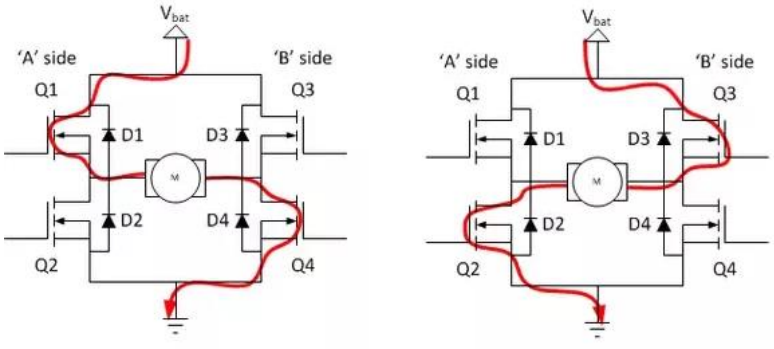


Slika 0-1 Mikrokontroler STM32F103C8T6 (Blue Pill)

Mikrokontroler se sastoji od procesora, memorije, 8 MHz kvarcni oscilatora, Micro USB konektora za napajanje, 4 pina za programiranje, tipkala reset te 48 pinova ulazno-izlaznih pinova. Pinovi se mogu koristiti kao GPIO pinovi, tajmeri ili pinovi za serijsku komunikaciju. Radni napon ovoga mikrokontrolera je od 2.0 do 3.6 V.

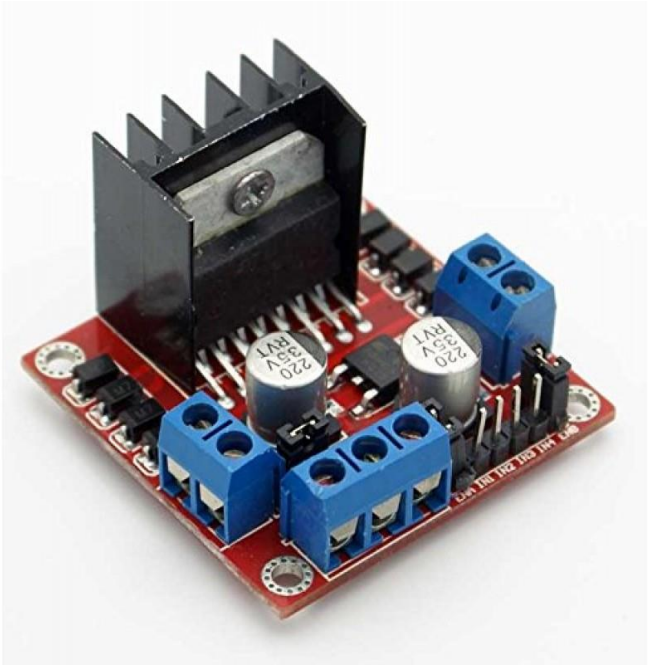
L298N modul za upravljanje motorima

Modul za upravljanje motorima koristimo kada imamo potrebu upravljati sa motorom tako da se on može pokretati u obje strane. Jednostavnom izvedbom, motor se preko 4 MOSFET tranzistora spaja na izvor napajanje.



Slika 0-2 Princip upravljanja motorom pomoću H-mosta

Ovim načinom izvedbe na jednostavan način se može upravljati, tako da pomoću tranzistora koji predstavljaju upravljive sklopke, propustimo struju kroz motor u jednom ili drugom smjeru ovisno u koju stranu želimo zakrenuti motor. Ovaj modul još nazivamo i dvojni H-most, zato jer omogućava istovremeno upravljanje s dva motora.



Slika 0-3 L298N modul za upravljanje motorima

Na sebi ima izvode koje koristimo za priključivanje motora i napona, te 4 pina koje koristimo za logičko upravljanje modula.

HC-SR04 Ultrazvučni senzor

Kod mjerenja udaljenosti koristi se ultrazvučni senzor, ondnosno modul za rangiranje (*eng. Ranging*) udaljenosti HC-SR04. Kako nam samo ime kaže, ovaj senzor korisiti ultrazvučne valove pri mjerenju. Omogućuje nam beskontaktno mjerenje udajenosti u rasponu od 2cm do 4m. Sastavljen je od odašiljča i prijamnika ultrazvučnih valova, i jednog kvarcnog oscilatora. Za upravljanje ima pinove *Trig* i *Echo,* te dva pia za napajanje

Slika 0-4 HC-SR04 Ultrazvučni senzor

Ako na *Trig* pin ovog modula pošaljemo visoki signal duljine barem 10µs, on će započeti mjerenje udaljenosti. Mjerenje započinje tako da odašilje 8 ciklusa zvučnih signala i čeka njihov povratak na prijamniku. Nakon odašiljanja postavi izlazni pin *Echo* na visoku razinu i drži ga tako dok prijamnik ne primi nazad poslanje zvučne signale. Mjerenjem duljine trajanja visoke razine *Echo* izlaza dobijemo vrijeme potrebno da se zvučni signal poslan sa odašiljača vrati na prijamnik. Udaljenost u centimetrima dobijemo iz formule:

Jednadžba 1 Računanje udaljenosti

$$\text{Udaljenost} = \text{VrijemeVisokeRazine} * \frac{BrzinaSvijetlosti}{2}$$

VrijemeVisokeRazine dobijemo mjerenjem, dok je BrzinaSvijetlosti jednaka 340m/s.

SG90 Servo motor

Servomotori su rotacioni ili linearni aktuatori koji se danas koriste u većini pokretnih alata ili strojeva. Jednostavni su za korištenje I implementaciju. U ovome projektnom zadatku se koriste mali servo motori *SG90*. On radi na naponu od 5V ima mogućnost zakretanja od 180°.

Slika 0-5 SG90 Servo motor

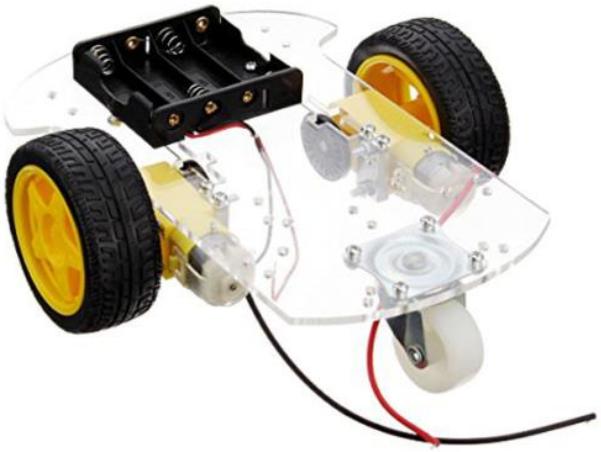
Za napajanje se koriste crvena i smeđa žica, dok se treća narandžasta koristi za upravljanje. S ovim servo motorom se upravlja pomoću PWM-a, točnije pulsno-širinske modulacije signala. Za upravljanje se korisit 50Hz frekvencija perioda 20ms. Za zakret motora na 0° potrebno je držati jedinicu 1ms od ukupnih 20ms. Za 180° jedinica stoji 2ms, a za sve ostale vrijednosti između 0° i 180° jedinica mora stajati 1 do 2 ms.

Robotska ruka

Slika 0-6 Robotska ruka

Istosmjerni motori s kotačima i platforma

Slika 0-7 Kotač i istosmjerni motor



Slika 0-8 Platforma s kotačima

Prilog 4. Programski kod mikrokontrolera

```
#include <Servo.h>

Servo clawServo, clawRotServo, topServo, botServo, baseServo, twinServo;
int pos2 = 90;
int in;
const int trigPin = PB14;
const int echoPin = PB15;
const int servoPin = PA8;

long duration;
int distance, pos, angle;
int closest = 100;
int EN_A = PA0;      //Enable pin for first motor
int IN1 = PC15;      //control pin for first motor
int IN2 = PC14;      //control pin for first motor
int IN3 = PC13;       //control pin for second motor
int IN4 = PA4;       //control pin for second motor
int EN_B = PA1;      //Enable pin for second motor
int motor_speed=120;
int motora_speed = motor_speed;
int motorb_speed = motor_speed;
int objPosAngle, objPosDistance;
Servo myservo;

void rotateLeft(int amount){
  int i;
  for(i = 0; i < amount; i++){
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    analogWrite(EN_A, motora_speed);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    analogWrite(EN_B, motorb_speed);
    delay(15);

  }
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

}


void rotateRight(int amount){
  int i;
  for(i = 0; i < amount; i++){
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    analogWrite(EN_A, motora_speed);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    analogWrite(EN_B, motorb_speed);
    delay(15);

  }
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
```

```
}

void moveFwd(int amount){
  int i;
  for(i = 0; i <= amount; i++){
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    analogWrite(EN_A, motora_speed);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    analogWrite(EN_B, motorb_speed);
    delay(45);

  }
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

}

int getDistance(){
  int distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH, 3500);
  distance= duration*0.034/2;
  return distance;
}

void findObject(int *posDistance, int *posAngle){
  int angle_sum = 0, angle_num = 0, dist = 200;
  for(pos = 50; pos <= 130; pos+=1){
    myservo.write(pos);
    delay(5);

    distance = getDistance();


    if (distance == 0){
      distance = 100;
    }
    if(dist < distance + distance*0.1 && dist > distance - distance * 0.1){
      angle_num++;
      angle_sum += pos;
     }else if(dist > distance){
      dist = distance;
      angle_num = 1;
      angle_sum = pos;

     }
  Serial1.print(distance);
  Serial1.print(",");
  Serial1.println(pos);
  }
  Serial1.println("as");

  for(pos = 130; pos >=50; pos-=1){
```

```
    myservo.write(pos);
    delay(5);

    distance = getDistance();
    if (distance == 0){
      distance = 100;
    }
    if(dist < distance + distance*0.1 && dist > distance - distance * 0.1){
      angle_num++;
      angle_sum += pos;
     }else if(dist > distance){
      dist = distance;
      angle_num = 1;
      angle_sum = pos;

     }
  Serial1.print(distance);
  Serial1.print(",");
  Serial1.println(pos);
  }
  Serial1.println("as");
  *posDistance = dist;
  *posAngle = angle_sum/angle_num;


}

void moveServo( int stopPos, Servo servo){
    int i;
    int startPos = servo.read();
    if(startPos < stopPos){
        for( i = startPos; i < stopPos; i++){
            servo.write(i);
            delay(15);
        }
    } else if (startPos > stopPos){
        for( i = startPos; i > stopPos; i--){
            servo.write(i);
            delay(15);
        }
    }
}



void grabObj(){
    moveServo(65, clawServo);
    delay(100);
    moveServo(180, baseServo);
    delay(100);
    moveServo(170, clawServo);
    delay(100);
    moveServo(45, topServo);
    delay(1000);
    moveServo(65, clawServo);
    delay(100);
    moveServo(90, topServo);
    delay(100);
    moveServo(80, baseServo);
    delay(100);
    moveServo(45, topServo);
    delay(100);
```

```
    moveServo(170, clawServo);
    delay(100);
    moveServo(90, topServo);
    delay(100);
    moveServo(65, clawServo);
    delay(100);
}


void setup ( ) {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(servoPin, OUTPUT);
  myservo.attach(servoPin);
  pinMode(EN_A, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(EN_B, OUTPUT);
  topServo.attach(PA2);
  delay(10);
  topServo.write(20);
  delay(100);
  moveServo(90, topServo);
  clawServo.attach(PA3);
  clawServo.write(90);
  baseServo.attach(PA6);
  baseServo.write(80);
  Serial1.begin(9600);
  void findObject(int *posDistance, int *posAngle);
}



void loop() {

  findObject(&objPosDistance, &objPosAngle);
  int rot = 90 - objPosAngle;


  if(objPosDistance <= 3){
    myservo.write(90);
    delay(1000);
    grabObj();
  }
  else if(objPosDistance < 65){
    if(rot < -5 - sqrt(objPosDistance)){
      rotateRight(sqrt(abs(rot)));
    }else if (rot > 5 + sqrt(objPosDistance)){
      rotateLeft(sqrt(abs(rot)));
    }else{
      moveFwd(objPosDistance/4);
    }

  }else{
    moveFwd(10);
  }



}
```

Prilog 5. Programski kod za iscrtavanje radara

```
import serial, time, pygame, math
from pygame.math import Vector2

white = (255,255,255)
black = (0,0,0)
red = (255,0,0)
green = (0,255,0)

vector = Vector2()



gameDisplay = pygame.display.set_mode((1000,800))
gameDisplay.fill(black)
x, y = 500, 750

pygame.draw.line(gameDisplay, white, (x, y), (x + vector[0], y + vector[1]))
pygame.display.flip()
dist = 500

angle_num = 0
angle_sum = 0
ser = serial.Serial('COM11', 9600)
time.sleep(2)
while(True):
    try:
        b = ser.readline()
        string_n = b.decode()
        string = string_n.rstrip()
        num = string.split(",")
        line_length = int(num[0]) * 3.5
        angle = 180 + int(num[1])
        if dist < line_length + 0.1 * line_length and dist > line_length - 0.1 * line_length:
            angle_num += 1
            angle_sum += angle
            print angle-180, line_length/3.5
        elif dist > line_length:
            dist = line_length
            pos = angle
            angle_num = 1
            angle_sum = angle
            print angle-180, line_length/3.5, 2

        vector.from_polar((200 * 3.5, angle))
        pygame.draw.line(gameDisplay, red, (x, y), (x + vector[0], y + vector[1]))
        vector.from_polar((line_length*2, angle))
        pygame.draw.line(gameDisplay, white, (x, y), (x + vector[0], y + vector[1]))
        pygame.display.flip()
    except:
        gameDisplay.fill(black)
        if angle_num == 0:
            vector.from_polar((200 * 4, pos))
        else:
            vector.from_polar((200 * 4, angle_sum / angle_num))
        pygame.draw.line(gameDisplay, green, (x, y), (x + vector[0], y + vector[1]))
        angle_num, angle_sum, dist = 0, 0, 500
```