SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 2474

Pretraživanje prostora korištenjem mobilne platforme Moway

Eugen Poljičak

Zagreb, lipanj 2012.

Zahvaljujem se prof. dr. sc. Stjepanu Bogdanu i asistentu Goranu Vasiljeviću za korisne prijedloge i pomoć prilikom izrade ovog rada.

Sadržaj

1. Uvod

Pretraživanje prostora zadatak je s kojim se susreću sva živa bića, od ljudi i životinja pa sve do raznih mikroorganizama. S tim zadatkom susrećemo se svakodnevno: traženje određenog predmeta, traženje lokacije gdje još nismo bili, potraga za plijenom kod životinja, potraga za partnerom, i slično. Procesi pretraživanja se pritom razlikuju: neki su kratkotrajni i jednostavni, a neki dugotrajni i kompliciraniji. U novije vrijeme dolazi do sve češće primjene robota, odnosno autonomnih sustava s ciljem kako bi se zamijenio ljudski rad u raznim aktivnostima. Primjerice, automatizirani sustavi proizvodnje u tvornicama obavljaju zadatke brže i točnije nego što bi to obavili ljudi; u medicini se koriste roboti malih dimenzija koji obavljaju zadaće na nedostupnim mjestima, postoje roboti koji mogu obavljati kućanske poslove, te isto tako postoje i roboti namijenjeni za pretraživanje prostora. Takvi roboti koriste se za pretraživanje opasnih i teško dostupnih mjesta (u slučaju požara, potresa, prirodne katastrofe), za pretraživanje većeg prostora (autonomne letjelice s kamerama koje snimaju prostor oko sebe), roboti zaduženi za razminiranje prostora, za traženje naftnih mrlja, za pretragu podvodnih lokaliteta, i slično. Pošto su takvi roboti najčešće autonomni, potrebno je osmisliti efikasan algoritam pretraživanja kako bi rezultati bili što bolji, usporedivi s inteligentnim, ljudskim pretraživanjem prostora.

Takvi algoritmi dijele se na determinističke i stohastičke. Deterministički, kao što i sama riječ kaže, su određeni, te je putanja robota unaprijed definirana (spiralni uzorak, cik-cak uzorak s ciljem prolaženja cjelokupnog prostora), gdje se pri kretanju kroz prostor često treba služiti podatcima sa senzora. S druge strane, stohastička putanja je neodređena, slučajna, te robot svaki put pretražuje prostor drukčijom slučajnom putanjom. Ljudi često ocijene determinističku putanju pogodnijom od stohastičke, mada se može pokazati i suprotno. Cilj ovog završnog rada jest usporediti efikasnost tih dviju putanja brojnim eksperimentima. Algoritmi pretrage ugraditi će se na Moway robotima čije su karakteristike opisane u nastavku rada. Područje pretraživanja bit će kvadrat stranica duljine 2 metra, a objekt kojeg će robot trebati pronaći bit će crni krug radijusa 20 cm. Među primjerima koji se koriste u praksi, ova pretraga slična je nalaženju naftne mrlje na vodenoj površini.

2. Mobilna platforma Moway

Moway je autonomni programabilni robot namijenjen izvođenju raznih praktičnih zadataka. Zbog svoje jednostavnosti izvrstan je za osobe koji prvi put dolaze u doticaj sa svijetom mobilnih robota, kao i za one koji već imaju dosta iskustva na području mobilne robotike, a žele proširiti svoje znanje i obavljati kompleksnije zadatke. Robot je opremljen brojnim senzorima koji mu pomažu pri gibanju u stvarnom svijetu. Sadrži i dva motora koji mu omogućavaju poprilično precizno kretanje po ravnim i glatkim podlogama. Svi ti periferni dijelovi spojeni su s glavnim mikrokontrolerom putem I2C/SPI komunikacijskih sabirnica. Robot također sadrži modul za proširenje, na kojeg se mogu priključiti bežični komunikacijski modul, video kamera ili prototip tiskane pločice na koju se mogu ugrađivati razni električni krugovi.

Zbog svojeg vanjskog dizajna i malih dimenzija (širina 8 cm i dužina 9 cm) Moway platforma je vrlo kompaktna, što utječe na sam vizualni efekt kojeg ostavlja svojim kretanjem. Na slici 1. prikazan je izgled Moway platforme te njezina veličina u stvarnom svijetu. Također, oblik robota omogućuje da se izbjegne mirovanje u slučaju kad se prilazi preprekama ili uglovima. Prikladno tome, Moway spada u kategoriju "džepnih robota". Moway je savršen uređaj za one koji žele naučiti nešto o minirobotici, kao i za one koji žele poboljšati svoje iskustvo na tom području. Čak i ako korisnik prvi put ulazi u svijet minirobotike, vrlo brzo će biti zadovoljan sa prvim postignutim rezultatima.

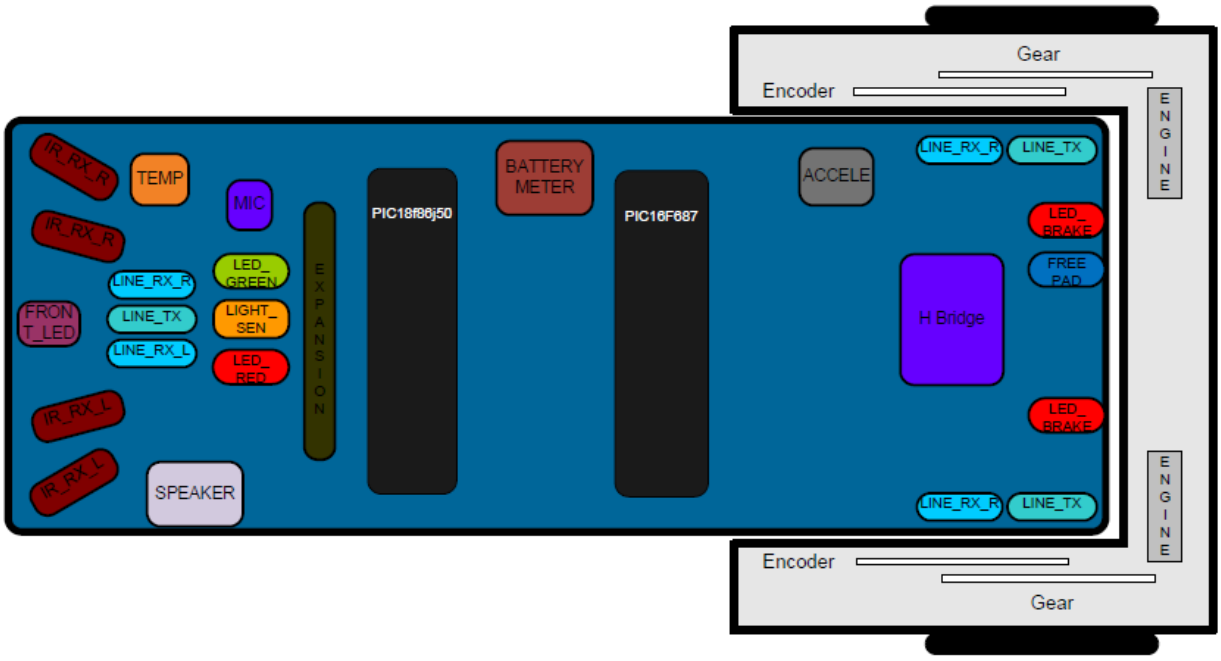
Slika 1: Usporedba veličine Moway platforme i obične žarulje

2.1. Sastavni dijelovi platforme Moway

U ovom poglavlju dan je opis dijelova od kojih se sastoji Moway. Za potrebe programiranja nije potrebno znati funkcioniranje svakog dijela robota, već samo određenih dijelova koji se koriste. Na slici 2. prikazan je položaj elemenata za upravljanje robotom, senzora, aktuatora i indikatora na tiskanoj pločici.

Unutar platforme možemo pronaći sljedeće elemente:

- Procesorska jedinica
- Pogonski sustav
- Sustav senzora i indikatora
- Izvor napajanja
- Modul za proširenje



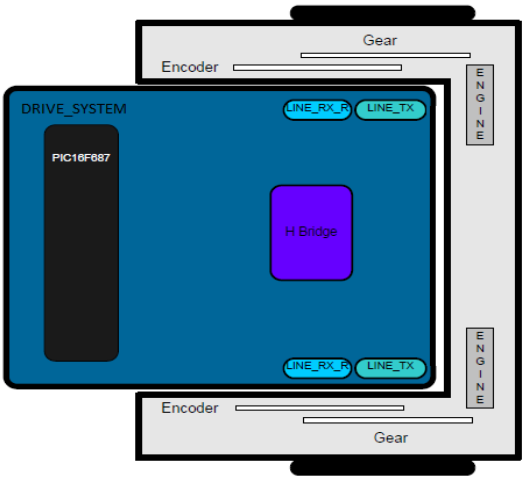
Slika 2: sastavni dijelovi Mowaya

2.2. Procesorska jedinica

Moway je upravljan PIC 18F86J50 mikrokontrolerom radne frekvencije 4 Mhz, proizvedenog u američkoj tvornici Microchip Technology. Na njegove ulaze i izlaze spojene su sve periferne jedinice (senzori, indikatori, aktuatori, modul za proširenje). Neke od njih zahtijevaju digitalni ulaz ili izlaz, neke zahtijevaju analogni ulaz ili izlaz, a neke su kontrolirane I2C/SPI komunikacijskim sabirnicama

2.3. Pogonski sustav

Moway za gibanje koristi grupu dvostrukih servo motora. Za njihovo upravljanje pogonski sustav se sastoji od elektroničkog i mehaničkog dijela (slika 3). Elektronički dio je uglavnom zadužen za kontroliranje brzine motora dok je mehanički dio zadužen za pokretanje motora, odnosno kotača.



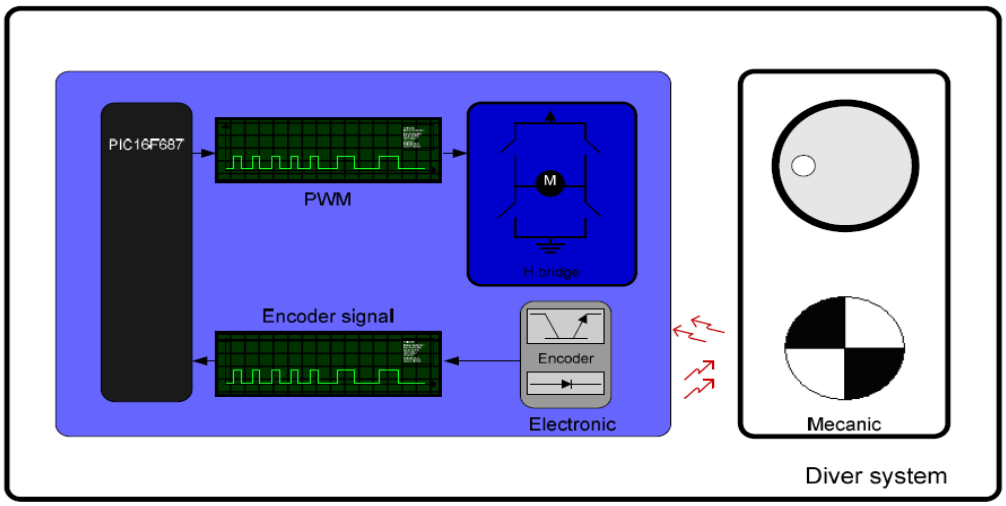
Slika 3: pogonski sustav, elektronički i mehanički

Pogonskim sustavom upravlja drugi mikrokontroler, PIC 16F687, koji se pri upravljanju motorima služi sljedećim parametrima:

1. Kontrola brzine – kontrolira se brzina svakog motora
2. Kontrola vremena – kontrolira vrijeme za svaku naredbu s preciznošću od 100 ms
3. Kontrola prijeđenog puta – kontrolira prijeđenu udaljenost za svaku naredbu s preciznošću od 1 mm
4. Generalni brzinomjer – računa ukupan prijeđeni put od početne naredbe
5. Kontrola kuta – kontrolira kut kad se Moway rotira

Glavni mikrokontroler (PIC 18F86J50) šalje putem I2C sabirnice naredbu za kontroliranje motora pogonskom sustavu koji kontrolira motore. Tu naredbu preuzima mikrokontroler pogonskog sustava (PIC 16F687), te se oslobađa glavni mikrokontroler kako bi on mogao nastaviti obavljati druge zadatke.

Kontrola brzine provodi se pomoću povratne informacije sa signala enkodera, kako je prikazano na slici 4. Informacija o rotaciji kotača stvara se očitanjem boje s naljepnice enkodera koja je spojena na kotač i infracrvenog senzora usmjerenog prema naljepnici: naljepnica sadrži crne i bijele dijelove, ta kada pokazuje crni dio, logički izlaz bit će '1', a kad pokazuje bijeli dio logički izlaz bit će '0'. Mikrokontroler analizira taj signal te mjerenjem širine impulsa (odnosno, mjerenjem trajanja logičkih stanja '0' i '1') može odrediti točnu brzinu okretanja svakog kotača. Moway će na taj način moći održati jednaku brzinu na bilo kojoj površini.



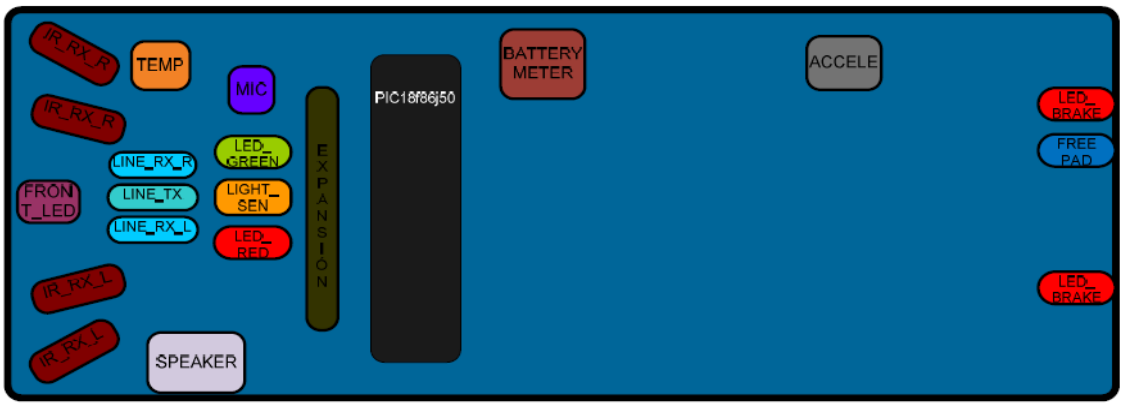
Slika 4: princip kontroliranja brzine motora

Kako bismo poslali naredbu za pokretanje motora, sve što trebamo napraviti jest upisati parametre gibanja u gotove funkcije namijenjene pokretanju i rotaciji motora. Te funkcije bit će opisane u poglavlju vezanom uz programiranje.

2.4. Sustav senzora i indikatora

Ovaj sustav sastoji se od grupa različitih senzora i indikatora (slika 5), spojenih na Moway-ov mikroprocesor, putem kojih robot ostvaruje interakciju s vanjskim svijetom:

- Linijski senzori
- Senzori za detekciju prepreke
- Senzor svjetlosti
- Senzor temperature
- Mikrofon
- Zvučnik
- Akcelerometar
- Napunjenost baterije
- LED diode



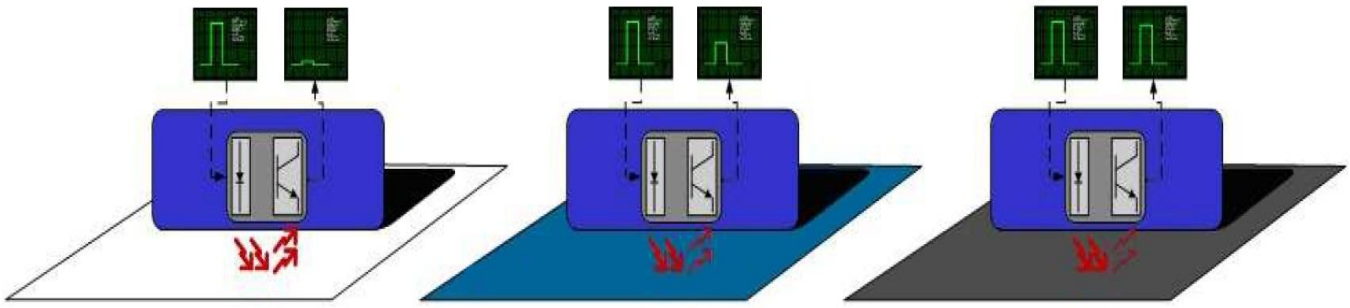
Slika 5: sustav senzora i indikatora

2.4.1. Linijski senzori

Slika 6: lokacija senzora linije

Ovi senzori su dva optička sprežnika smještena na prednjem dijelu platforme (slika 6), na donjoj strani tiskane pločice, kako bi bili usmjereni prema podlozi. Oni koriste refleksiju infracrvenog svjetla za određivanje refleksivnosti podloge na kojoj se robot nalazi. Na mikrokontroler su spojeni analognim ulazima kako bi se mogli detektirati snažni kontrasti na podlozi, primjerice granica bijelog i crnog područja. Izvor emitiranog svjetla i detektor smješteni su tako da emiter šalje snop infracrvene svjetlosti prema podlozi, a detektor prima reflektirani snop .

Linijski senzori u stvari ne mogu odrediti obojenost podloge, već postotak reflektirane svjetlosti. Ovisno o količini primljene svjetlosti (odnosno svjetlosti reflektirane od podloge), tranzistor detektora na svom izlazu zadržava niski, srednji ili visoki napon. Primjerice, svijetle površine reflektirati će većinu infracrvenog svijetla te ćemo stoga imati nizak napon na izlazu tranzistora. Obojene površine reflektiraju određen udio emitiranog svijetla, ostavljajući srednji iznos napona na tranzistoru, a tamne površine koje upijaju gotovo svu svjetlost a reflektiraju vrlo malo, ostaviti će visok napon na izlazu senzora. Na ovaj način se podatci sa senzora mogu čitati analogno, te korištenjem gotovih funkcija možemo očitati vrijednost osvijetljenosti podloge u vrijednostima od '0' do '255', gdje '0' označava jako svijetlu podlogu, a '255' jako tamnu podlogu. Može se dogoditi da za različite boje podloge dobijemo istu vrijednost reflektivnosti podloge, na primjer ukoliko testiramo bijeli papir i lakirani parket, sa senzora ćemo očitati vrijednosti u rasponu od '15' do '17'. Na slici 7. prikazani su iznosi napona na emiteru i detektoru, ovisno o reflektivnosti podloge.



Slika 7: iznos napona na emiteru i detektoru, ovisno o tipu podloge

2.4.2. Senzori za detekciju prepreke

Slično senzorima za praćenje linije, ovi senzori također koriste infracrveno svjetlo kako bi detektirali objekte locirane u neposrednoj blizini ispred platforme. Ovu grupu senzora čine dva emitera infracrvenog svjetla (jedan na lijevoj i jedan na desnoj strani platforme), te četiri detektora infracrvenog svijetla, po dva na svakoj strani platforme. Izlazi detektora spojeni su na analogne ulaze mikrokontrolera te se tako može odrediti prisutnost objekta u blizini (digitalni način rada), te se može izmjeriti udaljenost objekta (analogni način rada).

Princip funkcioniranja senzora sličan je funkcioniranju senzora za praćenje linije: emiter generira infracrveni snop svjetlosti trajanja 70 µs koji će se odbiti od prepreke i doći do detektora. Nakon što se signal elektronički obradi (odnosno prođe kroz analogno digitalni pretvornik), mikrokontroler ga može obraditi na analogni ili digitalni način. Ukoliko se obrađuje digitalnim načinom, udaljenost za koju se označava prisutnost objekta iznosi 3 cm, te funkcija vraća rezultat '1'. Analognim načinom rada detektiraju se objekti udaljeni do 10 cm te za njih funkcija vraća male vrijednosti (do '30'), a za bliske objekte vrijednost s funkcije biti će u intervalu '200' d0 '255'. Važno je napomenuti da je za točnije mjerenje udaljenosti preporučljiva svjetla okolina, kako bi se poboljšala refleksija infracrvenog svijetla o prepreku.

Ova četiri senzora nalaze se na prednjem dijelu tiskane pločice, te su usmjereni pod određenim kutevima u odnosu središnju simetralu. Tako postoje prednji lijevi i desni (pod kutevima ± 22.5°) te bočni lijevi i desni (pod kutevima ± 45°) senzori za detekciju prepreke, kao što je prikazano na slici 8.

Slika 8: lokacija senzora prepreke na prednjem dijelu robota

2.4.3. Senzor svjetlosti

Senzor svjetlosti mjeri intenzitet svjetlosti koja ulazi kroz otvor u obliku polumjeseca na prednjem dijelu platforme. Pošto otvor gleda prema naprijed, lagano se može utvrditi gdje je lociran izvor svjetlosti te skladno tome djelovati na određen način, ovisno o tipu zadatka. Izlaz svjetlosnog senzora također je spojen na analogni ulaz mikrokontrolera, te se može odrediti intenzitet svjetlosti čitanjem vrijednosti s analogno digitalnog pretvornika. Funkcija vraća vrijednosti iz intervala '0' do '255', gdje veća vrijednost znači jači intenzitet osvijetljenosti. Usporedbom prethodne i trenutne vrijednosti funkcije može se zaključiti u kojem smjeru se nalazi izvor svjetlosti.

2.4.4. Senzor temperature

Za mjerenje temperature Moway ima ugrađen termootpornik; odnosno poluvodič čiji se otpor smanjuje s porastom temperature. Termootpornik je spojen na analogni ulaz mikrokontrolera te se tako jednostavnim čitanjem vrijednosti struje pomoću analogno digitalnog pretvornika dobije vrijednost temperature u tom trenutku. Senzor se nalazi na prednjem dijelu tiskane pločice, pored desnog bočnog senzora za detekciju prepreke.

2.4.5. Akcelerometar

Akcelerometar (slika 9) je uređaj koji mjeri ubrzanje prilikom kretanja ili rotacije. Postoji više tipova akceleratora, od kojih je većina zasnovana na piezoelektričnim kristalima, no radi njihove veličine oni nisu uključeni u platformu Moway. Radi potrebe za manjim dimenzijama Moway koristi kapacitivni akcelerometar, koji se sastoji od dva kondenzatora vrlo malenih dimenzija, čiji se električni kapacitet mijenja s promjenom akceleracije. Mjerenjem vrijednosti akcelerometra za *X*, *Y* i *Z* os može se znati dali ja Moway pravilno pozicioniran, prevrnut ili nagnut.

Slika 9: akcelerometar i njegove osi

2.4.6. Mikrofon

Mikrofon omogućuje detektiranje zvukova frekvencija od 100 Hz do 20 kHz. Spojen je na analogni ulaz mikrokontrolera tako da može odrediti prisutnost zvuka (digitalni način rada) ili intenzitet zvuka čitanjem s analogno digitalnog pretvornika (analogni način rada).

2.4.7. Zvučnik

Zvučnik je spojen direktno na mikrokontroler, te može ispuštati tonove frekvencija od 250 Hz do 65 kHz.

2.4.8. Napunjenost baterije

Moway koristi litij - polimer punjivu bateriju, koja je spojena na analogni ulaz mikrokontrolera te se čitanjem vrijednosti s analogno digitalnog pretvornika može odrediti postotak napunjenosti baterije.

2.4.9. LED diode

Platforma sadrži nekoliko LED dioda: prednja dioda nalazi se na prednjoj strani motora, u sredini između senzora prepreke označenih na slici x. Na otvoru u obliku polumjeseca gdje je smješten senzor svjetlosti nalaze se dvije diode, crvena i zelena. Na stražnjoj strani platforme nalaze se dvije crvene diode, po jedna u blizini svakog kotača. Na slici 10. prikazan je Moway s upaljenim prednjim diodama (crvena, zelena i bijela), koje su isprogramirane tako da signaliziraju pronalazak objekta pretraživanja.

Slika 10: Moway s upaljenim prednjim diodama

2.5. Izvor napajanja

Moway koristi LiPo (litij - polimer) punjivu bateriju malih dimenzija. Baterija se može puniti putem USB kabela koji povezuje USB priključak kompjutera i MINI-USB-B priključak Moway platforme. Nije potrebno čekati bateriju da se potpuno isprazni kako bismo ju ponovno napunili jer LiPo ćelija pripada novoj generaciji baterija koje nemaju memorijski efekt. Trajanje napunjene baterije najviše ovisi o aktivnostima senzora i motora prilikom uključenosti robota, dok punjenje baterije traje oko 2 sata.

Sustav napajanja kontrolira dvije LED diode koje se nalaze na stražnjoj strani platforme, pored MINI-USB-B priključka. Zelena dioda svijetli ukoliko je Moway uključen, a narančasta svijetli ukoliko se baterija puni. Kad je baterija napunjena, narančasta dioda se gasi.

2.6. Modul za proširenje

Na modul za proširenje, koji se nalazi u središnjem dijelu platforme, mogu se priključiti sljedeći elementi:

- bežični komunikacijski RF modul, koji dolazi u kompletu s RFUsb-om
- video kamera
- prototip tiskane pločice na koju se mogu ugrađivati razni električni krugovi

2.6.1. RF modul i RFUsb

Radiofrekvencijski modul (skraćeno, RF modul) omogućuje komuniciranje s ostalim Moway platformama ili s kompjuterom, pritom koristeći RFUsb. Moway koristi BZI-RF2GH4 model RF modula, koji može ostvariti dvosmjernu komunikaciju s potvrdom prijenosa. Mogu se ostvariti dva slučaja, da RF modul spojen na Moway odašilje podatke, a RFUsb spojen na kompjuter prima podatke, i obrnuto. Komunikacija između RF modula i mikrokontrolera ostvarena je SPI sabirnicom. Glavne karakteristike BZI-RF2GH4 RF modula su:

- mala potrošnja
- radna frekvencija: 2-4 GHz
- snaga prijenosa između -18 i 0 dBm

- brzina prijenosa između 1 i 2 Mbps
- mogućnost selekcije 128 prijenosnih kanala od strane SPI sabirnice

Korisnik se ne treba brinuti o hardverskom dijelu ovog RF modula jer je sve već napravljeno, njegova dužnost je samo priključiti RF modul s modulom za proširenje kako je prikazano na slici 11, te programski uskladiti bežično komuniciranje s više Moway platformi ukoliko se radi o takvom zadatku. Ako je potrebno komunicirati sa samo jednim Moway-om, dovoljno je spojiti RF modul s platformom, te RFUsb s kompjuterom. RF modul se s platformom povezuje pomoću osam pinova: četiri su namijenjena za SPI sabirnicu, dva za kontrolu modula i dva za napajanje modula.

Slika 11: RF modul, RFUsb te RF modul priključen s modulom za proširenje

2.6.2. Video kamera

Pomoću modula za video kameru (Moway Camera Board) na slici 12, na kompjuteru se može prikazati slika koju Moway trenutno "vidi". Camera Board šalje slike bežičnim putem na uređaj Moway Videocap, kojeg je potrebno USB kabelom povezati na kompjuter kako bismo mogli gledati video od onog što Moway trenutno vidi. Za upravljanje kamerom, korisnik treba poznavati osnovne funkcije poput funkcija camera ON/OFF za paljenje i gašenje kamere, te treba odabrati isti prijenosni kanal u glavnom programu kao i u Moway Videocap-u. Modul s kamerom se povezuje s platformom na isti način kao i RF modul, pomoću osam pinova: četiri su namijenjena za SPI sabirnicu, dva za kontrolu modula i dva za napajanje modula.

2.6.3. Modul s prototipom tiskane pločice

Ovaj modul služi za razna proširenja koje korisnik želi dodati platformi, kao što je prikazano na slici 12. Potrebno je dizajnirati određeni električni krug, spojiti ga na tiskanu pločicu, zatim spojiti cjelokupni modul s tiskanom pločicom na modul za proširenje, te programirati funkcije koje je korisnik zamisli za tu pločicu. Modul je povezan s platformom pomoću osam pinova, na isti način kao i RF modul i modul za video kameru.

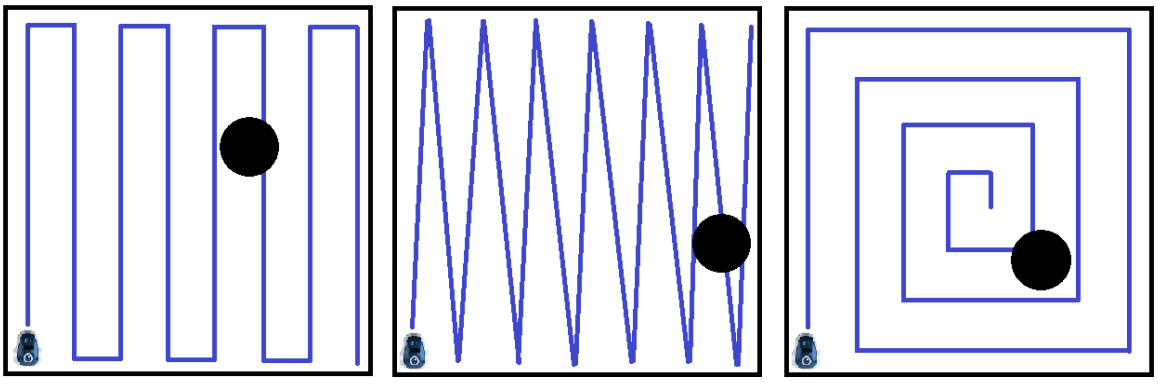
Slika 12: modul za video kameru (Moway Camera Board) te modul s tiskanom pločicom

3. Deterministički i stohastički princip pretraživanja prostora

Na prvi pogled, deterministički algoritam pretrage se čini efikasnijim od stohastičkog – sigurni smo da će robot pretražiti cijeli prostor jer smo mu unaprijed zadali takvu putanju, dok za stohastičko pretraživanje nismo toliko sigurni. Može se dogoditi da robot pretražuje jedan dio područja, zatim prijeđe u drugi dio te se opet vrati na početak, i tek nakon dosta proteklog vremena (odnosno, nakon puno pretraživanja istih, krivih područja) dođe do trećeg ciljanog područja u kojem se nalazi objekt pretraživanja. U takvom slučaju stohastičko pretraživanje trajalo bi daleko više vremena od determinističkog. No, može se dogoditi i da se slučajnom putanjom robot odmah uputi prema pravom objektu pretraživanja, te da stohastičko pretraživanje traje puno kraće od determinističkog. Dok smo za deterministično pretraživanje znamo koliko će otprilike trajati, za stohastičko to možemo samo nagađati. Zato treba provesti više eksperimenata kako bismo mogli uspoređivati ove dvije metode.

3.1. Deterministički princip pretraživanja

Kako je ovakvo pretraživanje određeno, a cilj je da se u potrazi za objektom obiđe cijeli prostor, robotu moramo zadati putanju kojom će se to ostvariti. Područje pretraživanja je kvadrat stranica duljine 2 metra, te se na takvom području mogu implementirati različiti tipovi putanja koje se neće isprepletati. Najpopularniji tipovi ovakvih putanja su spiralna i zig-zag putanja, kao što je prikazano na slici 13.



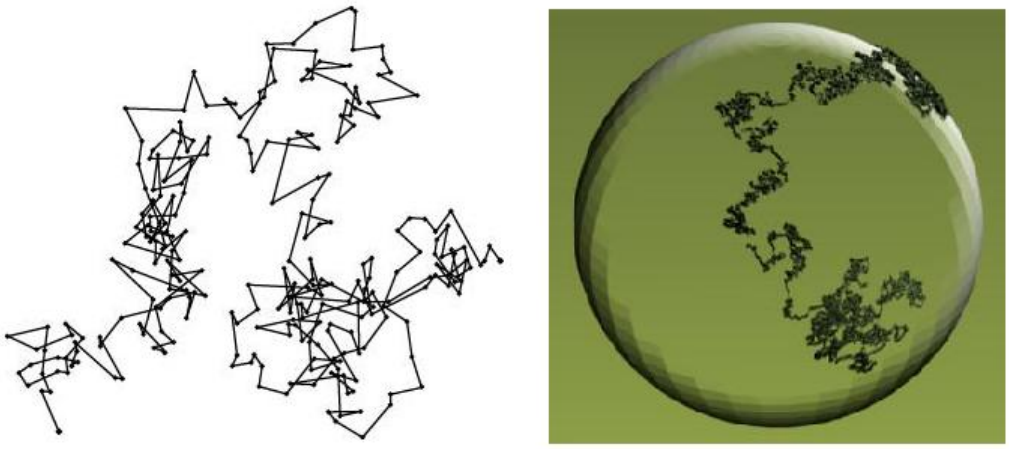
Slika 13:Pravokutna i oštrokutna zig-zag putanja te spiralna putanja zajedno s Moway-om i objektom pretraživanja. Plava linija predstavlja područje koje će pregledati linijski senzor

3.2. Stohastički princip pretraživanja

Generiranje stohastičke putanje kompliciranije je od generiranja determinističke putanje, te postaje puno zahtjevnije ukoliko želimo imati što raznolikiju putanju. Za ostvarenje gibanja stohastičkom putanjom Moway-u su potrebna dva parametra: kut za koji će se Moway zakrenuti i skok, odnosno pravocrtna udaljenost koju će zatim prijeći.

3.2.1. Brownovo gibanje

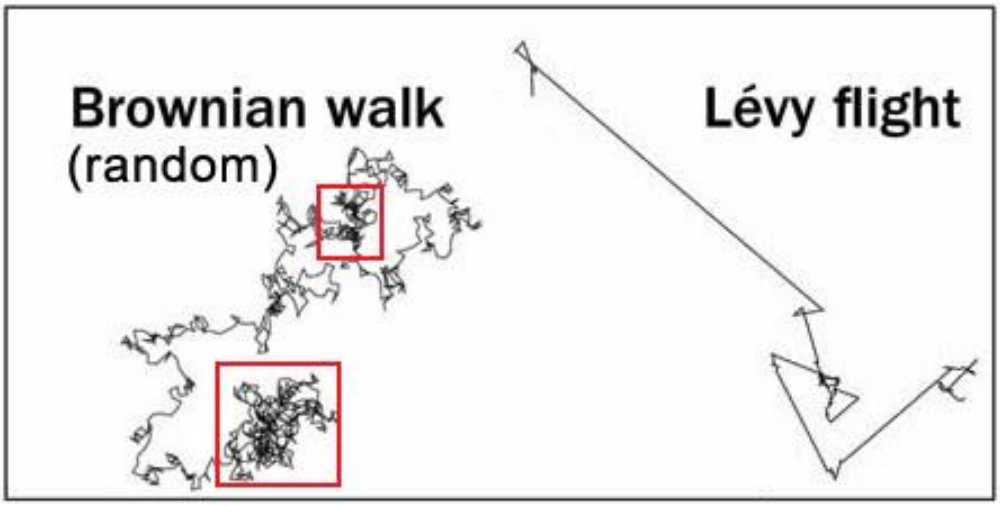
Najjednostavniji oblik stohastičke putanje jest proces gdje je kut raspoređen po uniformnoj razdiobi, odnosno jednaka je vjerojatnost da će se Moway zakrenuti za bilo koji kut unutar intervala (0°, 360°), te je iznos skoka pritom konstantan. Ovaj način gibanja zove se Brownovo gibanje prema škotskom botaničaru Robertu Brownu; on je promatrao gibanje čestica peludi unutar kapljice vode i gibanje čestica ugljenove prašine na površini alkohola te je uočio ovakvo kaotično gibanje. Prikaz gibanja može se vidjeti na slici 14. U novije doba ovakvo gibanje je primijećeno i pri gibanju molekula plina. Nedostatak ovakvog načina pretrage je njegova neefikasnost: naime, može se dogoditi da će robot tijekom gibanja često prelaziti već obiđeno područje, što utječe na ukupno vrijeme pretrage. To se dešava poglavito iz razloga što su mu skokovi jednaki i svako je novo područje pretraživanja vrlo blizu prethodnom, te ukoliko se jednolikom razdiobom generira kut blizak kutu od 180° robot će se opet vratiti na prethodno područje. Ovakvi slučajevi mogli bi se izbjeći pamćenjem prošlih nekoliko stanja te ''zabranom'' generiranja određenog intervala kuta kako se robot ne bi vraćao na već pregledano područje.



Slika 14: Grafički prikaz Brownowog gibanja, te mikroskopski prikaz gibanja čestica peludi unutar kapi vode

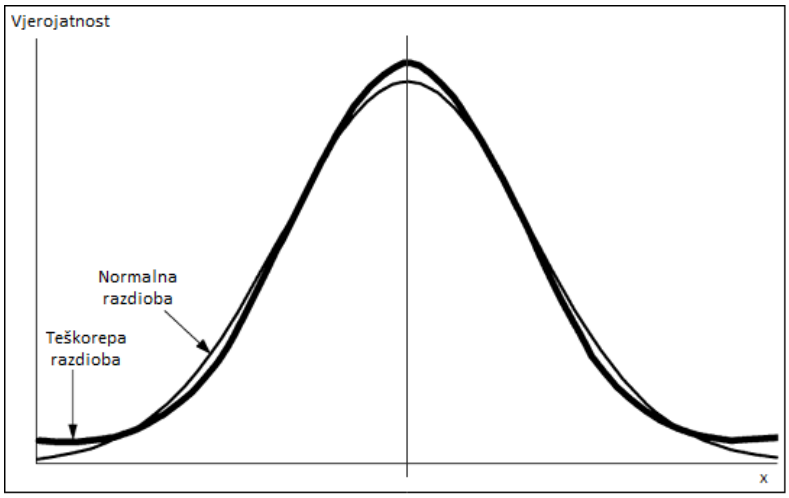
3.2.2. Levyjev let

U Brownowom gibanju skok je bio konstantan dok je kut bio određen stohastički; sljedeći korak je osmisliti putanju koja će koristiti stohastički generirana oba parametra, kut i skok. Ideja za ovakvo slučajno gibanje također potječe iz prirode; uočeno je da se neke životinje u potrazi za hranom kreću upravo ovakvom putanjom, sa slučajnim kutom i slučajnim skokom. Biolog Nicholas Pade je uočio da morski psi i drugi morski grabežljivci potpuno promijene model kretanja kad u oceanu ne mogu naći plijen u odnosu kad plijena ima dovoljno. Napuštaju Brownovo gibanje te ga zamjenjuju gibanjem pod nazivom Levyev let, kojeg karakterizira mješavina dugačkih skokova i puno manjih, kraćih pomaka u slučajnim smjerovima, kao što je prikazano na slici 15. Ovo otkriće bilo je fascinirajuće jer je slučajno gibanje primijećeno ne samo kod čestica, već i kod živih bića. Kompjuterski modeli su pokazivali na to da je Levyev let optimalan tip pretraživanja za grabežljivce u područjima s malo plijena, te da takva putanja maksimizira šansu slučajnog susreta s plijenom. Prvotna istraživanja u stvarnom svijetu bila su neuvjerljiva te dovela u pitanje ovu teoriju. Međutim, kasnije istraživanje u lipnju 2010. godine bilo je najveće dotad po broju prikupljenih podataka. Istražitelji su analizirali 13 milijuna kretnji koje su snimljene u preko 5700 dana prateći radio navigacijom 55 jedinki morskih životinja iz 14 različitih vrsti. Kako su se životinje kretale iz područja s obiljem plijena prema području s malo plijena, jednadžbe koje su opisivale njihovo kretanje sve više su opisivale Levyev let, a sve manje Brownovo gibanje.



Slika 15: Usporedba putanja Brownovog gibanja i Levyjevog leta. Crvenim kvadratima označeno je područje koje se kod Brownovog gibanja obilazi više puta

Francuski matematičar Paul Pierre Lévy bavio se proučavanjem stohastičkog gibanja te je po njemu nazvan oblik putanje Levyev let (eng. Levy flight). Levyjev let je slučajan hod kod kojeg duljine koraka imaju teškorepu (eng. Heavy-tailed) razdiobu vjerojatnosti. Teškorepa razdioba jest razdioba vjerojatnosti čiji krajevi nisu eksponencijalno omeđeni, što znači da je vjerojatnost za krajnje vrijednosti puno veća nego kod normalne (Gaussove) razdiobe. Usporedba tih dviju grafova vidi se na slici 16. Nakon što se odrede iznosi koraka u dimenziji većoj od jedan, naprave se skokovi u smjerovima koji su određeni uniformnom razdiobom (odnosno, jednaka je vjerojatnost za bilo koji smjer).



Slika 16: Graf normalne i teškorepe razdiobe

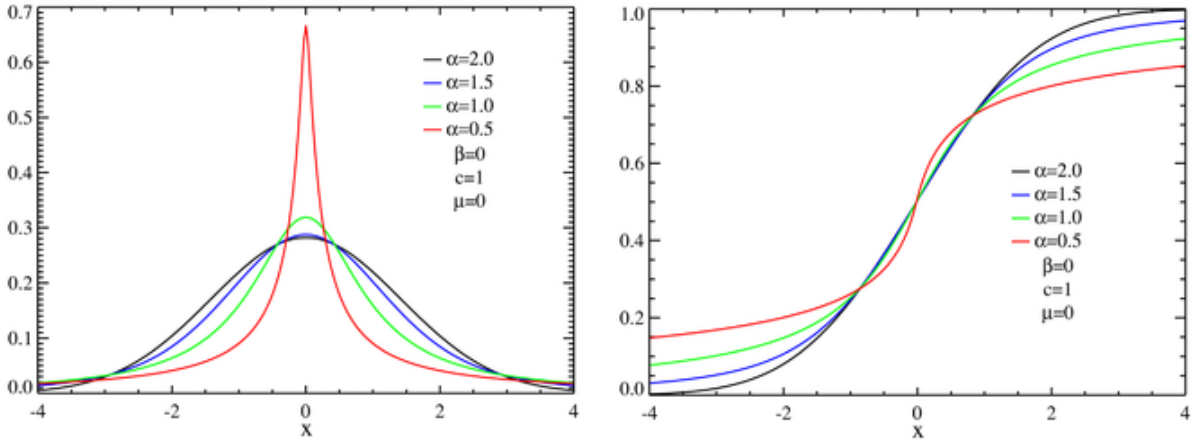
Levyev let po konstrukciji spada u Markovljeve procese, što znači da razdioba vjerojatnosti za buduće stanje procesa ovisi samo o trenutnom stanju, a ne o nizu događaja iz prošlosti. Iznosi skokova određuju se prema stabilnoj razdiobi: slučajna varijabla ima stabilnu razdiobu ukoliko linearna kombinacija dviju nezavisnih varijabli slučajne razdiobe ima također slučajnu razdiobu. Levyeva alfa-stabilna razdioba je jedna od familija stabilnih razdioba, uz normalnu i Cauchyjevu. Takve razdiobe mogu se opisati pomoću četiri parametra, μ, c, β i α:

- μ - lokacijski parametar, on ukazuje na pomak grafa razdiobe ulijevo ili udesno. Može se pronaći u lokacijsko-skalirajućim familijama razdioba, odnosno funkcija gustoće vjerojatnosti s obzirom na parametar μ imat će sljedeći izgled:

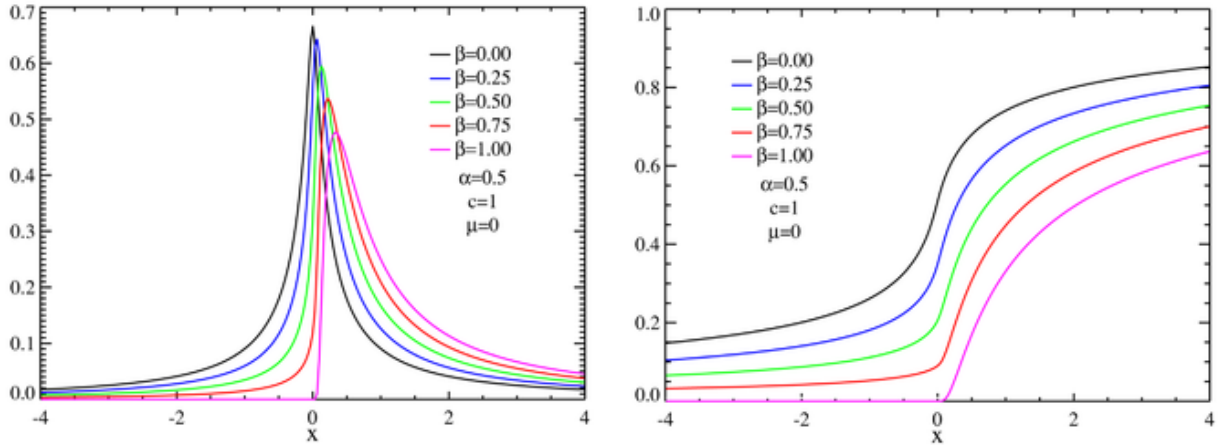
$$f_{\mu,\theta}(x) = f_{\mu}(x - \mu)$$

- c - parametar skaliranja, koji opisuje raširenost razdiobe vjerojatnosti; ukoliko je c velik razdioba će biti raširenija, a ukoliko je c malen razdioba će biti koncentrirana na manje područje
- α, β – parametri oblika razdiobe (eng. shape parameters). Takav parametar je bilo koji parametar razdiobe vjerojatnosti koji nije niti lokacijski niti skalirajući parametar. Ovaj parametar mora utjecati na oblik razdiobe, za razliku od gornja dva parametra koji posmiču distribuciju ulijevo ili udesno te ju rastežu ili skupljaju. Parametar $\alpha, \alpha \in (0,2]$, zove se parametar stabilnosti, te on pokazuje kakva će biti stabilnost gustoće razdiobe slučajne varijable x. Što je alfa veći to će razdioba biti stabilnija, odnosno protezati će se kroz manji interval, i obrnuto. Parametar $\beta, \beta \in [-1, 1]$ zove se parametar asimetričnosti, te govori koliko će se razlikovati gustoća razdiobe za interval $x > 0$ u odnosu na gustoću razdiobe za interval $x < 0$. Za $\beta = 1$ razdioba će postojati samo za vrijednosti $x > \mu$, a za $\beta = -1$ razdioba će postojati samo za vrijednosti $x < \mu$.

Na slici 17. prikazana je gustoća stabilne razdiobe i razdioba slučajne varijable x uz različite vrijednosti parametra stabilnosti α, a na slici 18. prikazana je gustoća stabilne razdiobe i razdioba slučajne varijable x uz različite vrijednosti parametra asimetričnosti β.



Slika 17: gustoća razdiobe (lijevo) i razdioba slučajne varijable x (desno) za pomak 0, jedinično skaliranje, bez asimetrije, uz različite parametre stabilnosti razdiobe α



Slika 18: gustoća razdiobe (lijevo) i razdioba slučajne varijable x (desno), uz pomak 0, jedinično skaliranje, uz koncentriranost 0.5, te uz različite parametre asimetrije razdiobe β

Kako bismo shvatili princip generiranja Levyevog leta potrebno je imati matematičku podlogu kako bismo shvatili odakle potječu brojne jednadžbe potrebne za ovaj zadatak. Levyjeva razdioba je jedna od rijetkih razdioba koje su stabilne i koje imaju funkciju gustoće vjerojatnosti koja se može zapisati analitički.

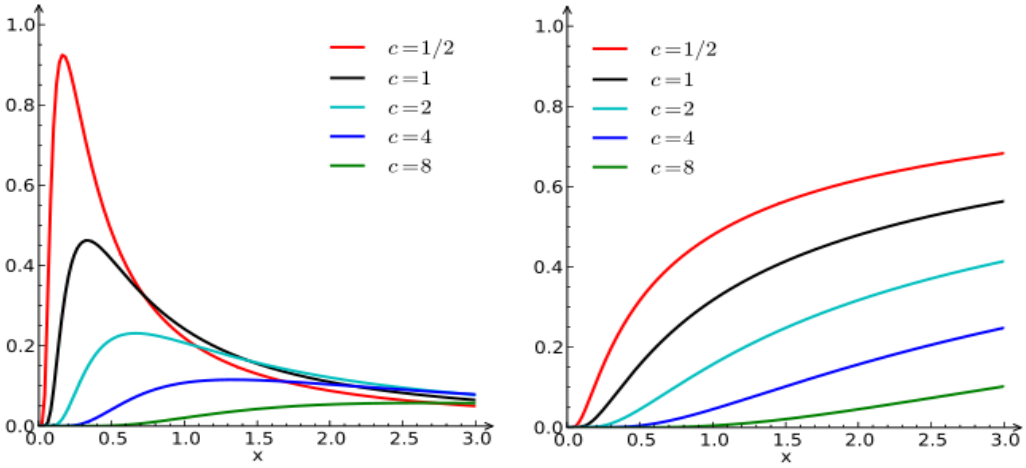
Funkcija gustoće razdiobe vjerojatnosti za Levyjevu razdiobu nad područjem $x \geq \mu$ jest

$$f(x;\mu,c) = \sqrt{\frac{c}{2\pi}}\frac{e^{-\frac{c}{2(x-\mu)}}}{(x-\mu)^{3/2}},$$

gdje je μ lokacijski parametar, a c je parametar skaliranja. Funkcija razdiobe slučajne varijable x jest $F(x;\mu,c) = erfc\left(\sqrt{c/2(x-\mu)}\right)$, gdje je funkcija $erfc(x)$ poznata pod nazivom Gaussova funkcija kvantila, a računa se kao:

$$erfc\,(x) = \frac{1}{2\pi}\int_x^\infty e^{-t^2/2}dt.$$

Na slici 19. prikazana je gustoća Levyjeve razdiobe vjerojatnosti te funkcija Levyjeve razdiobe slučajne varijable x uz različite parametre skaliranja c.



Slika 19: gustoća Levyjeve razdiobe vjerojatnosti (lijevo) te funkcija Levyjeve razdiobe slučajne varijable x (desno), uz različite parametre skaliranja c

Kako bismo kompjuterski generirali stabilne slučajne varijable uzimajući u obzir parametre μ, c, β i α, koristit ćemo Chambers-Mallows-Stuck metodu. Za lokacijski parametar uzima se $\mu = 0$, što znači da se razmatraju samo pozitivne vrijednosti slučajne varijable x, a za parametar skaliranja uzima se $c = 1$, odnosno jedinično skaliranje koje znači da razdioba neće biti niti raširenija na veće niti koncentriranija na manje područje. Koristeći metodu možemo generirati stabilnu varijablu $X \sim S_\alpha(1, \beta, 0)$, za $\alpha \in (0,2]$ i $\beta \in [-1, 1]$:

- najprije je potrebno generirati jednoliko raspoređenu varijablu V unutar intervala $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ te slučajnu varijablu W eksponencijalne razdiobe s parametrom $\lambda = 1$. Varijablu W računamo po formuli $W = -\log W'$, gdje je W' jednoliko raspoređena varijabla na intervalu $[0,1]$.
- Za $\alpha \neq 0$ računamo

$$X = S_{\alpha,\beta} \cdot \frac{\sin\left(\alpha\left(V + B_{\alpha,\beta}\right)\right)}{\left(cos(V)\right)^{1/\alpha}} \cdot \left(\frac{cos\left(V - \alpha\left(V + B_{\alpha,\beta}\right)\right)}{W}\right)^{(1-\alpha)/\alpha},$$

gdje je

$$B_{\alpha,\beta} = \frac{\arctan\left(\beta \tan\frac{\pi\alpha}{2}\right)}{\alpha},$$

$$S_{\alpha,\beta} = \left[1 + \beta^2 tan^2 \frac{\pi\alpha}{2}\right]^{1/(2\alpha)}$$

- Za $\alpha = 1$ računamo

$$X = \frac{2}{\pi}\left[\left(\frac{\pi}{2} + \beta V\right)\tan V - \beta \log\left(\frac{\frac{\pi}{2} W \cos V}{\frac{\pi}{2} + \beta V}\right)\right].$$

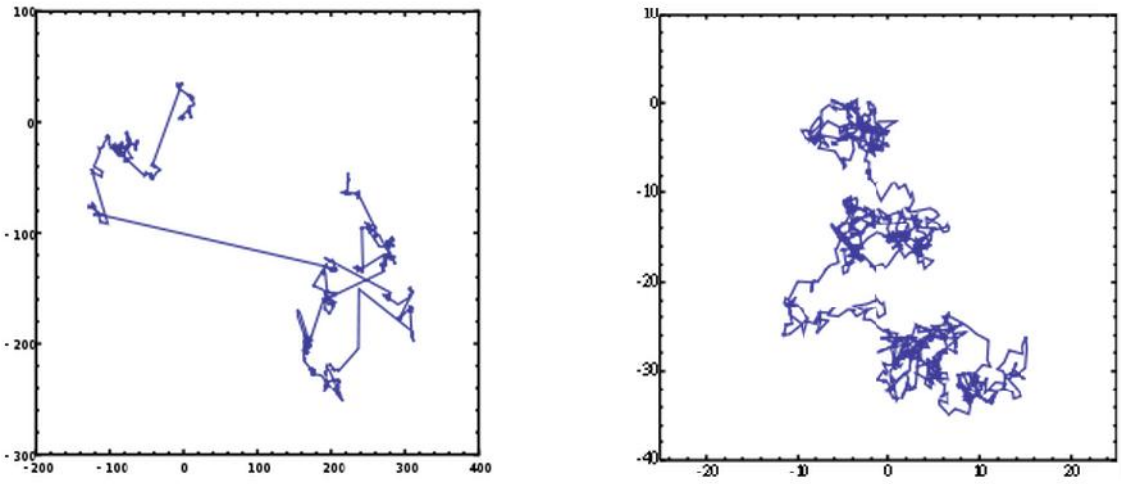
Ovisno o različitim parametrima α i β mogu se dobiti različite stabilne razdiobe:

- $\alpha = 2$ i $\beta = 0$ za normalnu, Gaussovu razdiobu
- $\alpha = 1$ i $\beta = 0$ za Cauchyjevu razdiobu
- $\alpha = 1/2$ i $\beta = 1$ za Levyjevu razdiobu
- $\alpha = 1$ i $\beta = 1$ za Landauovu razdiobu
- $\alpha = 3/2$ i $\beta = 0$ za Holtsmarkovu razdiobu.

Za potrebe zadatka korištena je Levyjeva razdioba, te se formula za izračun koraka za tu razdiobu može pojednostaviti na:

$$X(0.5, 1) = \left[4W\ sin^2\left(\frac{1}{2\left(V - \frac{\pi}{2}\right)}\right)\right]^{-1}.$$

Na slici 20. prikazane su simulacije Levyjevog leta i Brownovog gibanja kroz 1000 koraka.

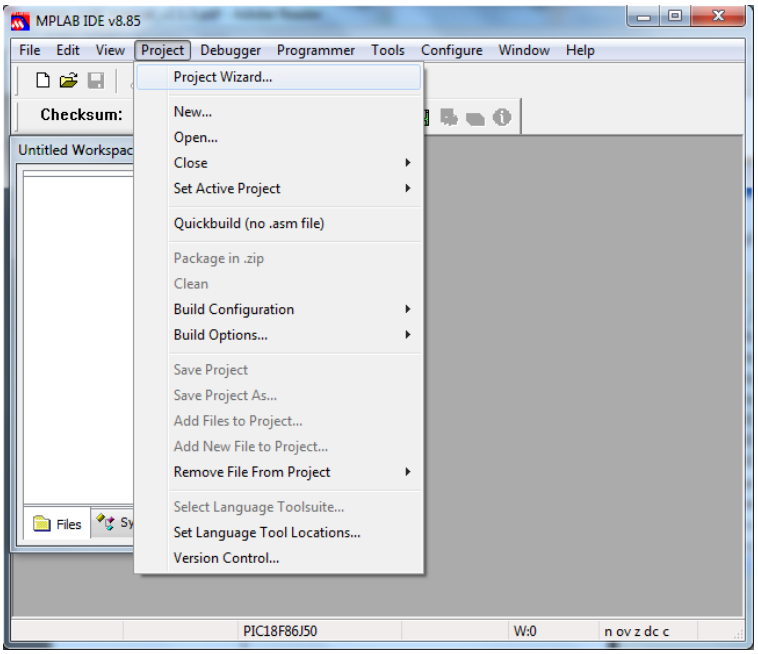


Slika 20: Primjer Levyevog leta (lijevo) i Brownovog gibanja (desno) kroz 1000 koraka. Početak gibanja nalazi se u koordinati [0,0]. Kutna usmjerenost je uniformno distribuirana, a duljina koraka za Levyjev hod distribuirana je prema Levyjevoj razdiobi s parametrima $\alpha = 1/2$ i $\beta = 1$. Duljina koraka za Brownovo gibanje distribuirana je prema normalnoj razdiobi, odnosno $\alpha = 2$ i $\beta = 0$.

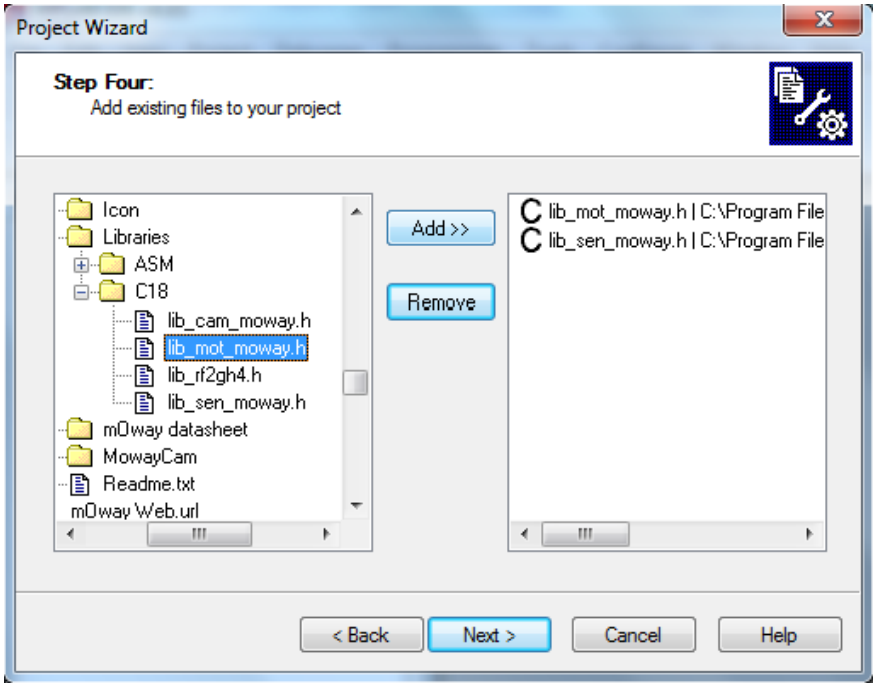
4. Programiranje

“Mozak“ Moway platforme jest PIC18F86J50 mikrokontroler, kojega je potrebno isprogramirati kako bi obavljao željenu zadaću. Osnovni jezik mikrokontrolera je asembler i svaki procesor im svoj. Programiranje u asembleru je vrlo zahtjevno te su programi često dugački i nepregledni. Zato se u pravilu koristi jedan od viših programskih jezika, uglavnom C, koji se zatim prevodi u asemblerski jezik za željeni tip mikrokontrolera. Tako programer ne mora voditi računa o posebnostima pojedinog mikrokontrolera. Osnovni alat za programiranje PIC mikrokontrolera je MPLAB IDE program, uz kojeg treba instalirati C18 kompajler kako bi se omogućilo prevođenje C jezika u asembler. C18 također sadrži mnoštvo ugrađenih funkcija koje pomažu pri programiranju mikrokontrolera (I2C, SPI, ...), te je veličina generiranih programa veća nego kod asemblera. Ovaj programski paket idealan je za brz napredak u programiranju mikrokontrolera te za izvođenje srednje teških zadataka na Mowayu.

Pokretanjem MPLAB IDE-a otvara se glavni prozor u kojem treba odabrati naredbu za kreiranje novog projekta. Otvorit će se čarobnjak za izradu projekta (slika 21), gdje treba odabrati mikrokontroler te kompajler koje ćemo koristiti, odnosno PIC18F86J50 mikrokontroler te MPLAB C18 C Compiler. Nadalje, potrebno je odabrati lokaciju na disku gdje želimo spremati projekt, te je u idućem koraku omogućeno dodavanje već postojećih datoteka u projekt, poput programskih biblioteka koje sadrže funkcije za upravljanje Mowayem i očitavanje vrijednosti s njegovih brojnih senzora. Takve datoteke poznate su pod nazivom header datoteke, a za komuniciranje s Mowayom treba uključiti *lib_sen_moway.h*, *lib_mot_moway.h* i *p18f86j50.h* datoteke, odnosno biblioteke s funkcijama vezanim uz senzore i motore, te biblioteku za PIC 18F86J50 mikrokontroler (slika 22). Nakon što se kreira projekt unutar njega se mogu kreirati nove datoteke, kao i što se mogu uključivati nove biblioteke (konkretno, biblioteke delays.h te p18f86j50.h koje služe za definiranje vremenskih stanki te za uspješno programiranje mikrokontrolera).



Slika 21: odabir čarobnjaka za izradu projekta



Slika 22: unošenje određenih biblioteka i ostalih tipova datoteka u projekt

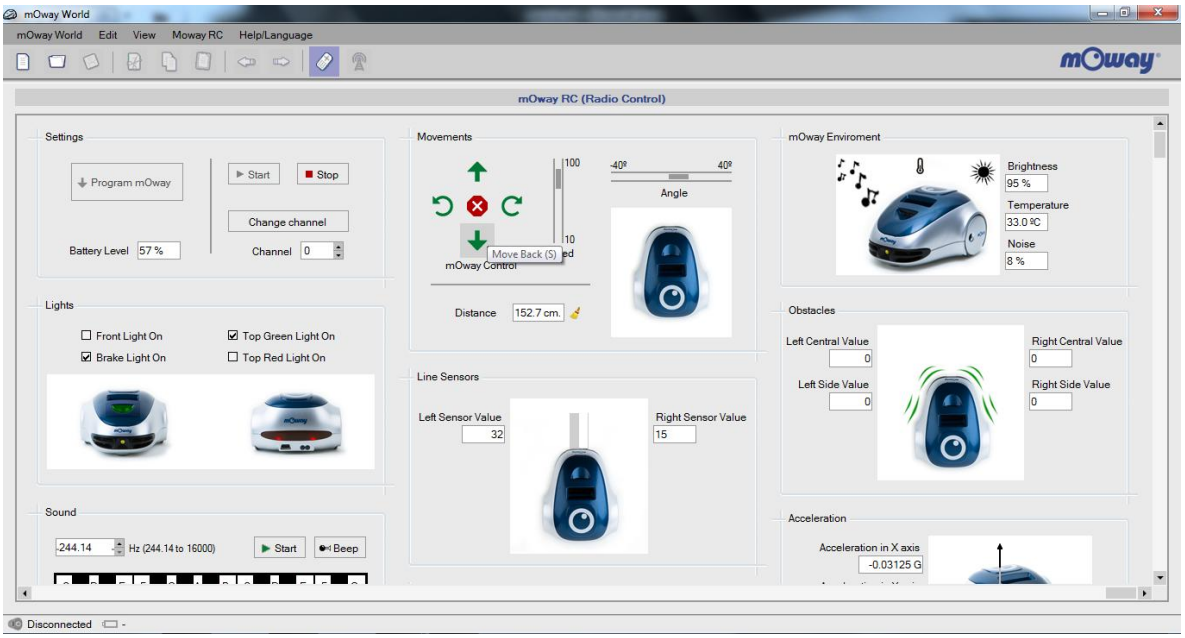
Za programiranje zadatka potrebno je stvoriti novu C datoteku te u nju upisati programski kôd. Na početku kôda potrebno je uključiti sve biblioteke koje smo prethodno dodali u projekt naredbom #include. Zatim je potrebno u kod ubaciti dio za reakciju prilikom reseta ili prekida programa. Taj dio koda je obavezan, a može se naći na službenim internet stranicama Mowaya. Zatim slijedi programiranje deterministične, odnosno stohastičke putanje.

4.1. Programiranje determinističke putanje

Determinističke zig-zag putanje jednostavnije su za izvesti od spiralnih putanja; potrebno je samo isprogramirati segment putanje od jedne granice prostora pretraživanja do nasuprotne granice i natrag, te taj segment ponavljati sve dok Moway ne naiđe na objekt pretraživanja (crni krug). Kako bi Moway mogao autonomno izvesti ove operacije ukoliko ga postavimo u kvadratno područje, trebat će se služiti podacima sa senzora. Kako se Moway prilikom kretanja kroz prostor kreće po različitim površinama, idealno je iskoristiti podatke s linijskih senzora koji se nalaze na donjoj strani platforme, usmjereni prema tlu. Prilikom prolaženja kroz te površine linijski senzori prate refleksivnost podloge na kojoj se nalaze te ovisno o podlozi vraćaju vrijednosti u rasponu '0' (za jako svjetlu podlogu koja reflektira svu svjetlost)do '255' (za jako tamnu podlogu koja upija svu svjetlost). Za očitanje vrijednosti sa senzora linije koristimo funkciju SEN_LINE_ANALOG(LINE_L) za lijevi, odnosno SEN_LINE_ANALOG(RIGHT_L) za desni linijski senzor. Ove funkcije su ugrađene u *lib_sen_moway.h* biblioteku. U stvarnom području na kojem će se provoditi eksperiment Moway će tražiti objekt pritom prolazeći preko tri tipa podloge:

- lakirani parket (za kojeg senzori linije vraćaju vrijednosti manje od '25'), odnosno glavna podloga preko koje će Moway najviše prolaziti
- izolir traka tamnoplave boje zalijepljena na parket (za koju senzori linije vraćaju vrijednosti u rangu od '50' do '160'), odnosno granica područja pretraživanja
- crni krug promjera 20 cm (za kojeg senzori linije vraćaju vrijednosti veće od '200')

Važno je napomenuti da se ovakve vrijednosti ne mogu očekivati od svake Moway platforme, jer senzori istog tipa često daju drukčije vrijednosti za istu stvar, odnosno senzore je potrebno je kalibrirati. Testiranje senzora jednostavno se izvodi pomoću programskog alata MowayWorld: potrebno je priključiti Moway mini-usb kabelom za kompjuter, upaliti MowayWorld te otvoriti MowayRC prozor. Program će automatski zahtijevati reprogramiranje Mowaya kako bi se omogućila bežična komunikacija. Zatim je potrebno priključiti RF modul za platformu, te RFUsb za kompjuter. Tako je ostvarena bežična komunikacija između programa i platforme. Sljedeći korak je paljenje platforme te testiranje senzora, indikatora i motora putem Moway RC (Radio Control) centra, čije je sučelje prikazano na slici 23.



Slika 23: Izgled sučelja Moway RC centra prilikom uključenog Mowaya bežični spojenog na kompjuter. U prozorima se mogu vidjeti vrijednosti određenih senzora

Pomoću Moway centra mogu se izmjeriti vrijednosti koje će senzori sa svakog Mowaya vratiti za određeni tip podloge. Tako, za primjer, linijski senzori s obje platforme vraćaju vrijednosti manje od '20' za lakirani parket, te vrijednosti veće od '200' za crni krug. Međutim, vrijednosti koje vraćaju za tamnoplavu granicu područja (izolir traku) se razlikuju: jedna platforma vraća vrijednosti od '50' do '100', a druga vrijednosti od '90' do '160'. Taj raspon vrijednosti treba uzeti u obzir prilikom prebacivanja programa na pojedinu platformu. Kako je ciljni objekt pretraživanja najtamniji objekt (crni krug), primijećen je jedan problem prilikom izvođenja eksperimenta u stvarnom svijetu: Moway je prilikom dolaska na granicu crnog kruga reagirao kao da je došao na granicu, odnosno ponašao se kao da ju želi izbjeći te je krenuo u suprotnom smjeru potpuno zanemarivši crno područje ispred njega. Do tog problema je dolazilo zato su se vrijednosti s linijskih senzora prilikom prilaska crnom krugu penjale između vrijednosti oko '20' (lakirani parket) do vrijednosti '200' (crni krug), no te vrijednosti se nisu trenutno promijenile (nije bilo trenutnog prelaska s '20' na '200'), već su se unutar tog malog vremenskog intervala pojavile vrijednosti unutar intervala '50' do '160', odnosno, glavni program koji se neprestano izvodi u kratkim ciklusima prepoznao je te vrijednosti kao granicu područja, te se Moway počeo tako i ponašati. Problem je riješen tako da je stavljen dodatan uvjet: ukoliko linijski senzori pokazuju vrijednosti za tamnoplavu izolir traku, Moway se treba

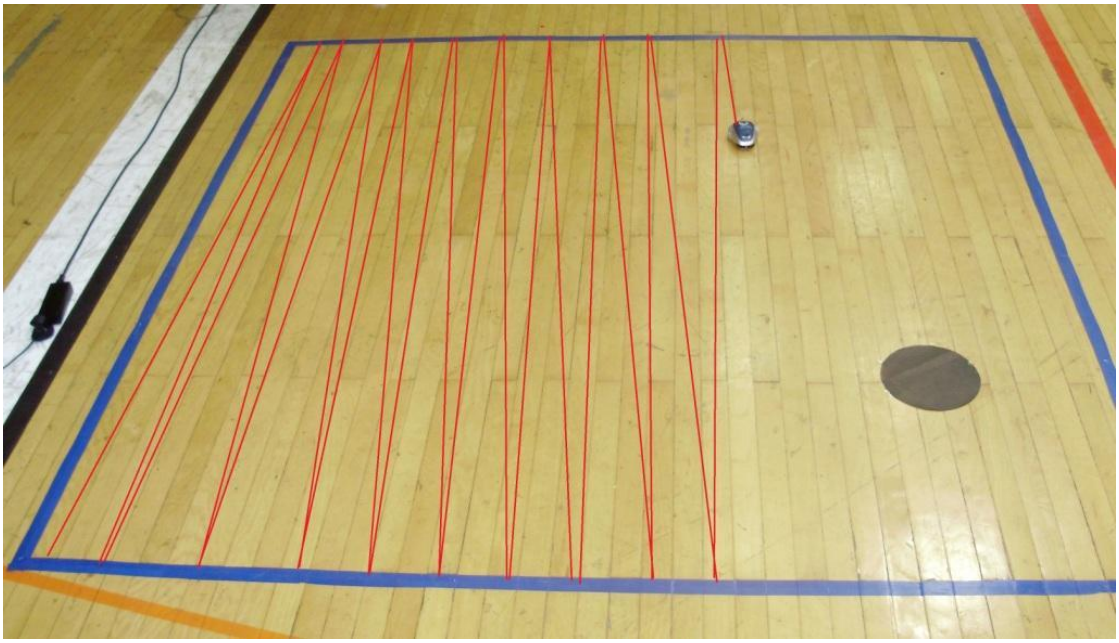
gibati ravno za udaljenost od 5 mm i provjeriti vrijednosti s linijskih senzora, te ukoliko je zabilježena vrijednost iznad '200' Moway se treba zaustaviti što označava kraj pretrage. Debljina trake je 3 cm, a promjer kruga 20 cm, tako da Moway neće izaći izvan područja pretraživanja pri ovom koraku. Ukoliko su nakon ovog pomaka očitane vrijednosti i dalje manje od '200', što znači da je Moway došao na granicu ali ne i na crni krug, Moway se zakreće u suprotnom smjeru te se nastavlja gibati ravno do iduće granice. Na ovaj način Moway svojim kratkim pomakom od 5 mm zauzme vrijeme koje je dovoljno da linijski senzori izračunaju točnu vrijednost refleksivnosti podloge. Ovi zadaci napisani su kao void funkcije uključene u glavni program, zato što će se često ponavljati pa je na ovaj način programski kod pregledniji.

Sljedeći korak, nakon što je definirano ponašanje robota u slučaju dolaska na granicu ili crni krug, jest programiranje determinističke putanje. U zadatku će se koristiti zig-zag tip determinističke putanje, zato što spiralna putanja neće biti toliko precizna. Spiralna putanja je zamišljena tako da Moway postavimo u centar područja, te da se on kreće ravnim skokovima koji se uzastopno povećavaju, pritom se rotirajući za kut od 90°, uvijek u istu stranu. Problem je što aktuatori na platformi nisu toliko precizni, te će se ponekad dogoditi da se Moway za kut iz intervala (88°, 92°), te ovakav odmak od nekoliko stupnjeva dosta utječe na izobličenje putanje. Spiralna putanja ima najviše takvih zakreta, odnosno u stvarnosti će imati najviše izobličenje, pa je zbog toga odbačena kao opcija. Pravokutna zig-zag putanja ima manje zakretanja od 90°, ali još uvijek puno više nego oštrokutna putanja, što će značiti da će izobličenje oštrokutne putanje biti najmanje. Provedeno je nekoliko eksperimenata kako bi se utvrdilo koji je tip putanje bolji; pokazano je da je oštrokutna puno točnija.

Algoritam za programiranje oštrokutne determinističke putanje jest sljedeći: prilikom pokretanja eksperimenta Moway postavimo u donji lijevi kut kvadratnog područja. Oštrokutna putanja isprogramirana je tako da Moway na početku krene ići ravno, sve dok ne naiđe na granicu (ili na crni krug, u kojem slučaju će se zaustaviti). Kad naiđe na granice, zarotira se udesno za 180° te se krene gibati naprijed sve dok ne naiđe na granicu ili crni krug. Naredba zakretanja za 180°bi značila da se Moway vraća istim putem kojim je krenuo, međutim testiranja su pokazala da se za zadani zakret od 180° Moway u stvarnom svijetu zakrene za nešto manje od 180°, odnosno baš za kut kakav je potreban.

Nakon rotacije, ako Moway stigne do druge granice, zakreće se ulijevo za 176.4°. Eksperimenti su pokazali da se prilikom ove naredbe Moway zakrene u lijevu stranu za kut manji od 180°, kao što smo i tražili. Treba napomenuti da funkcija za rotiranje Mowaya kao argument kuta prima cjelobrojni broj u rasponu od '0' do '100', koji je linearno skaliran s raspona (0°, 360°). To znači da će se za argument '49' Moway zakrenuti za $49 \cdot 3.6° = 176.4°$, odnosno to je prvi kut rotacije manji od 180° kojeg se može zadati Mowayu. Gornje opisani postupak označava kretanje Mowaya tijekom jednog segmenta, a kako se taj segment treba ponavljati sve dok se ne pronađe crni krug, najjednostavnije je njegov programski odsječak staviti unutar glavne while(1){} petlje u glavnom programu.

Prilikom programiranja deterministčkih putanja valja voditi računa o tome da robot svojim gibanjem obiđe cijeli prostor te da ne zaobiđe područje u kojem se nalazi crni krug. To znači da međusobna udaljenost između linija pretraživanja ne smije biti veća od promjera kruga, jer se u protivnom može dogoditi da se crni krug nalazi točno između dvije linije te da ga linijski senzor prilikom pretraživanja neće uočiti. Ovaj problem je riješen tako što se Moway prilikom svakog kontakta s granicom zakreće za kut malo manji od 180°, što osigurava prilično detaljan prolazak kroz cijelo područje. Izgled takve putanje na stvarnom eksperimentu prikazan je na slici 24.



Slika 24: Prikaz oštrokutne putanje na stvarnom eksperimentu. Može se primjetiti da pojedine susjedne linije nisu skroz paralelne; to se događa zbog nesavršenosti aktuatora

4.2. Programiranje stohastičke putanje

Za razliku kod determinističke putanje, gdje je glavni cilj bio da Moway prijeđe cijeli prostor pretraživanja, te gdje su se javljali problemi odstupanja od zamišljene putanje radi nesavršenosti aktuatora, kod stohastičke putanje takvi "nedovoljno precizni" zakreti neće predstavljati problem, jer je takvo gibanje ionako kaotično.

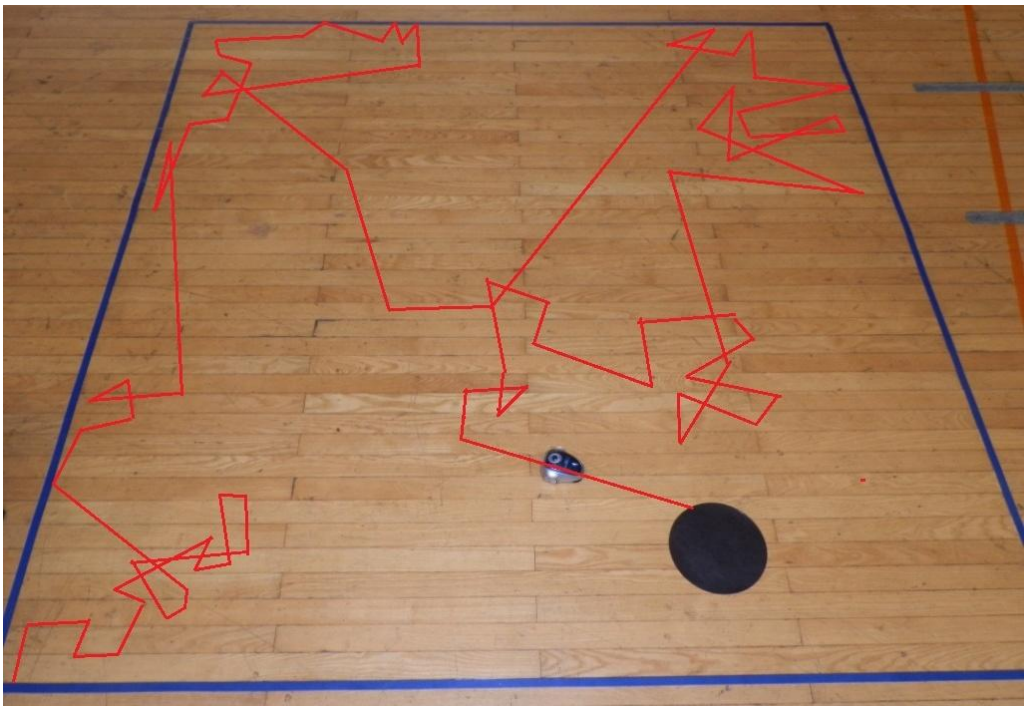
Treba spomenuti da je kod stohastičke putanje primijenjen drugačiji princip razlikovanja tamnoplave granice i crnog kruga; dok se kod determinističke putanje Moway u slučaju nailaska na granicu gibao ravno 5 mm te tada ponovno provjeravao tip podloge, kod stohastičke će se primijeniti precizniji tip takve provjere, gdje se Moway neće gibati ravno, već će se zakretati. Opis reagiranja u toj situaciji je sljedeći: ukoliko linijski senzori pokazuju vrijednosti za tamnoplavu izolir traku, Moway se treba zarotirati udesno za 36° te provjeriti vrijednosti s linijskih senzora, te ukoliko je zabilježena vrijednost iznad '200' Moway se treba zaustaviti što označava kraj pretrage. Ukoliko su tom rotacijom pronađene vrijednosti manje od '200' Moway se zakreće ulijevo za 72° te ponovno provjerava vrijednosti za prisutnost crnog objekta. Ukoliko ni tada nije pronađen crni objekt, Moway se vraća u početni položaj dodatnom rotacijom za 36° udesno te se nastavlja s izvođenjem programa, odnosno zakretanjem u suprotnim smjerovima kako bi se udaljio od granice. Ovaj način provjera tipa okolne podloge je točniji nego kod determinističke putanje: tamo se može dogoditi da Moway prođe uz sam rub crnog kruga, prepozna najprije vrijednosti za granicu, te krene naprijed 5 mm i time napusti rub kruga. Pretraživanje neće biti uzaludno jer će prilikom sljedećeg skretanja od granice Moway naići na taj krug, samo što će to zahtijevati dodatno vrijeme. U takvom slučaju bi rotiranje ulijevo i udesno kao kod stohastičke putanje bilo idealno, ali pošto bi se rotiranje obavljalo prilikom svakog nailaska na granicu, a svako rotiranje nije dovoljno precizno, dogodilo bi se da će se Moway nakon nekoliko takvih iteracija početi gibati dijagonalno po području, a ne paralelno s granicama. Zato je kod determinističkog gibanja ostao tip provjere područja koji je jednostavniji i manje točan, ali ne zakreće putanju te je povoljniji za izgled putanje, što je važniji faktor.

Nakon što je programski sređeno točno prepoznavanje podloge, može se započeti programirati putanja. Za stohastičko gibanje potrebno je generirati kut zakreta dobiven

uniformnom razdiobom, te skok dobiven Levyevom razdiobom. Nakon što se izračunaju ti parametri, robotu se zadaje da se zakrene za dobiveni kut, te da se nakon toga giba ravno naprijed za udaljenost dobivenog skoka. Takav segment se neprekidno ponavlja unutar glavne while(1){} petlje, sve dok robot ne naiđe na crni krug. Za vrijeme izvođenja jednog segmenta gibanja uključene su funkcije koje provjeravaju prisutnost granice prostora, odnosno crnog objekta. U slučaju da Moway dođe do granice, prekida se izvođenje segmenta te se provjerava dali je Moway došao na granicu ili crni krug. Ukoliko je došao na granicu koja se nalazila s lijeve strane, Moway se zarotira za 126° udesno i obrnuto, te se tada započinje izvođenje novogeneriranog segmenta.

Naredbe za rotaciju i pravocrtno gibanje Mowaya nalaze se unutar *lib_mot_moway.h* biblioteke. Funkcija za rotiranje Mowaya kao argument kuta prima cjelobrojni broj u rasponu od '0' do '100', koji je linearno skaliran s raspona (0°, 360°), a također može primiti i vrijeme rotiranja u rasponu od 0 do 25.5 sekundi, koje se funkciji zadaje kao cjelobrojni broj u rasponu od '0' do '255', odnosno razlučivost je desetinka sekunde. Funkcija za pravocrtno gibanje kao argument skoka može primiti udaljenost ili vrijeme gibanja, koji joj se zadaju kao cjelobrojni broj u rasponu od '0' do '255', s tim da je razlučivost za udaljenost 1 mm, a za vrijeme desetinka sekunde. To znači da će najveći argument za udaljenost za broj '255' predstavljati 25.5 cm, što nije zadovoljavajuće za zadatak jer Levyjev let u svojoj putanji sadrži povremene velike skokove, veće od 25.5 cm. Zato se kao argument za pravocrtno gibanje uzima vrijeme gibanja, a ne udaljenost. Moway je isprogramiran da putuje sa 100° nazivne brzine, te je provedeno par mjerenja kako bi se zaključila njegova maksimalna brzina, koja iznosi $v = 15.1\ cm/s$. Nakon što program formulom za izračun Levyjeve slučajne varijable generira iznos skoka, potrebno je taj iznos pretvoriti u vrijeme gibanja, što se radi formulom $vrijeme = \frac{skok}{brzina} \cdot 10$, gdje je $vrijeme$ ukupan broj desetinka sekundi koji se zaokružuje na cijeli broj te kao takav zadaje funkciji za pravocrtno gibanje. Maksimalna vrijednost skoka koja se može zadati Mowayju jest $d = v \cdot t_{max} = 15.1 \cdot 25.5 = 385.05\ cm$, što je i više nego dovoljna udaljenost za područje pretraživanja $2 \times 2\ m^2$. Levyjeva razdioba ponekad generira prevelik skok, puno veći od duljine 2 m, te je takvu vrijednost potrebno reducirati. Reduciranje je obavljeno modulo operacijom duljine koraka s brojem 130, što znači da je maksimalna dozvoljena vrijednost skoka u području pretraživanja jednaka 130 cm.

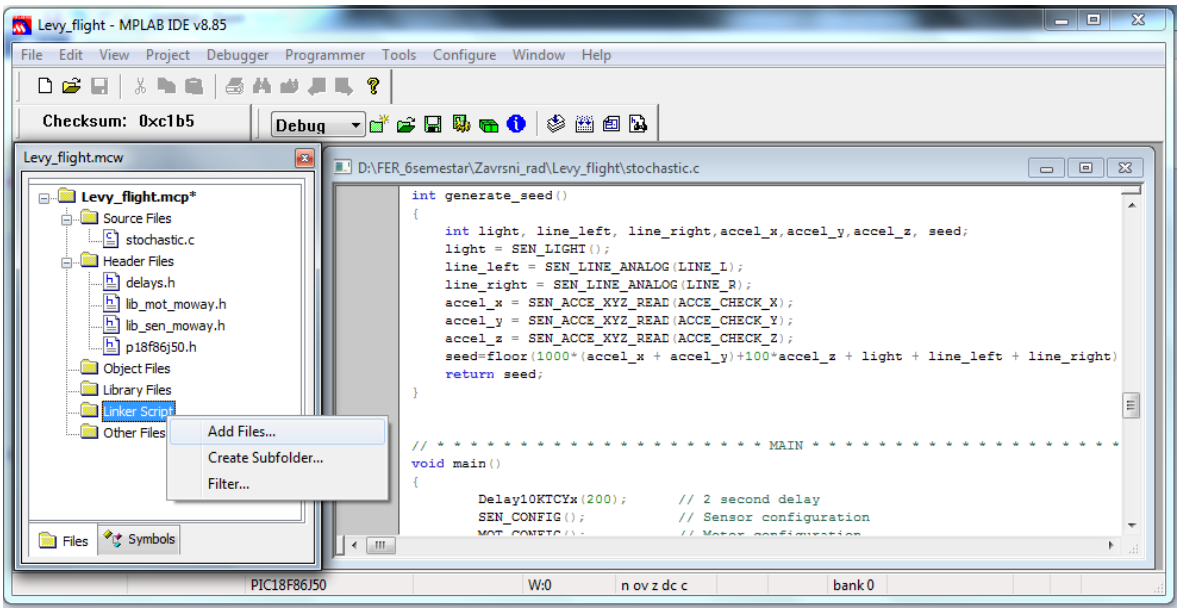
Za dobivanje slučajnih vrijednosti kuta i skoka potrebno je koristiti funkcije generiranja preudoslučajnih brojeva. Funkcija *rand ()* će prilikom izvođenja uvijek generirati isti niz pseudoslučajnih brojeva, što bi značilo da će se Moway prilikom svakog uključivanja uvijek kretati istom slučajnom putanjom, što nije dobro riješenje. Ovakav problem se u pravilu rješava umetanjem *srand ((unsigned) time (NULL))* linije koda, koja omogućava dobivanje različitih nizova pseudoslučajnih brojeva, te bi se na taj način Moway prilikom svakog uključivanja kretao drukčijom putanjom. Funkcija *srand (seed)* kao argument *seed* uzima broj sekundi proteklih nakon ponoći, 1. Siječnja 1970. Problem se nalazi u tome što PIC mikrokontroler nema svoj unutarnji sat koji mjeri vrijeme, te se ovakav program neće htjeti kompajlirati. Zato treba na osmisliti novi način kako će Moway prilikom svakog novog pokretanja koristiti različiti *seed* da bi se generirali različiti nizovi pseudoslučajnih brojeva, odnosno različite putanje. Prvotna ideja je bila da se u unutarnju memoriju mikrokontrolera na određenu adresu upiše vrijednost '1'. Zatim bi program za stohastičku putanju na početku izvođenja pročitao vrijednost s te lokacije, te ju koristio kao *seed* u *srand (seed)* funkciji, a pri kraju izvođenja programa bi tu vrijednost uvećao za 1 i spremio natrag na istu adresu. Tako bi svakim uključenjem Moway koristio različiti *seed* za generiranje pseudoslučajnih brojeva. Međutim, ni ovu ideju nije moguće realizirati pošto PIC 18F86J50 mikrokontroler ne omogućava upisivanje podatka u internu memoriju. Konačno rješenje problema jest generiranje *seed*-a na temelju vrijednosti očitanih s različitih senzora: sa senzora svjetlosti, s oba linijska senzora, te sa sva tri akcelerometra. Dok senzori svjetlosti i linijski senzori vraćaju cjelobrojne vrijednosti u intervalu '0' do '255', akcelerometri vraćaju decimalne vrijednosti puno manje od 1, te je njihove vrijednosti potrebno pomnožiti s velikim brojem kako bismo dobili vrijednosti sumjerljive s vrijednostima ostalih senzora. Formula za računanje konačnog *seed*-a jest: *seed = floor(1000*(accel_x + accel_y)+100*accel_z + light + line_left + line_right);* gdje na kraju dobivamo cjelobrojni *seed* koji se koristi u *srand(seed)* funkciji kako bi se dobio različiti niz pseudoslučajnih brojeva. Akcelerometar za Z os vraća oko deset puta veće vrijednosti od akcelerometra za X i Y os, te se zato množi s manjim faktorom. Vrijednosti s akcelerometra su puno promjenljivije od vrijednosti s ostala tri senzora, te se na ovaj način Moway svakim novim uključenjem krene gibati drukčijom putanjom, što je bio cilj. Izgled putanje “Levyjev hod“ na stvarnom eksperimentu prikazan je na slici 25.



Slika 25: Prikaz putanje Levyjevog hoda na stvarnom eksperimentu. Vide se područja s puno manjih koraka koja je Moway detaljnije pretražio, te duži skokovi u ostala područja

Nakon što je napisan kompletan kod u MPLAB IDE programu, uz uključene biblioteke potrebno je još uključiti takozvanu “linker“ skriptu. Ona opisuje kako bi dijelovi ulaznih datoteka trebali biti mapirani u izlaznu datoteku, te služi za kontrolu memorijske raspodjele za izlaznu datoteku. Nju se dodaje u projekt desnim klikom na kategoriju “Linker Script“ u glavnom prozoru MPLAB IDE-a, te se odabere naredba “Add Files“, i zatim se odabere lokacija te datoteke na disku kako bi se ona unijela u program. Skripta je dostupna na službenoj internet stranici Mowaya, u okviru primjera nekoliko programa za Moway. Na slici 26. prikazan je princip uključivanja “linker“ skripte u projekt, a taj princip je isti i za ostale tipove datoteka (header datoteke, datoteke s programskim kodom, bibliotekama, objektima i ostale vrste datoteka).

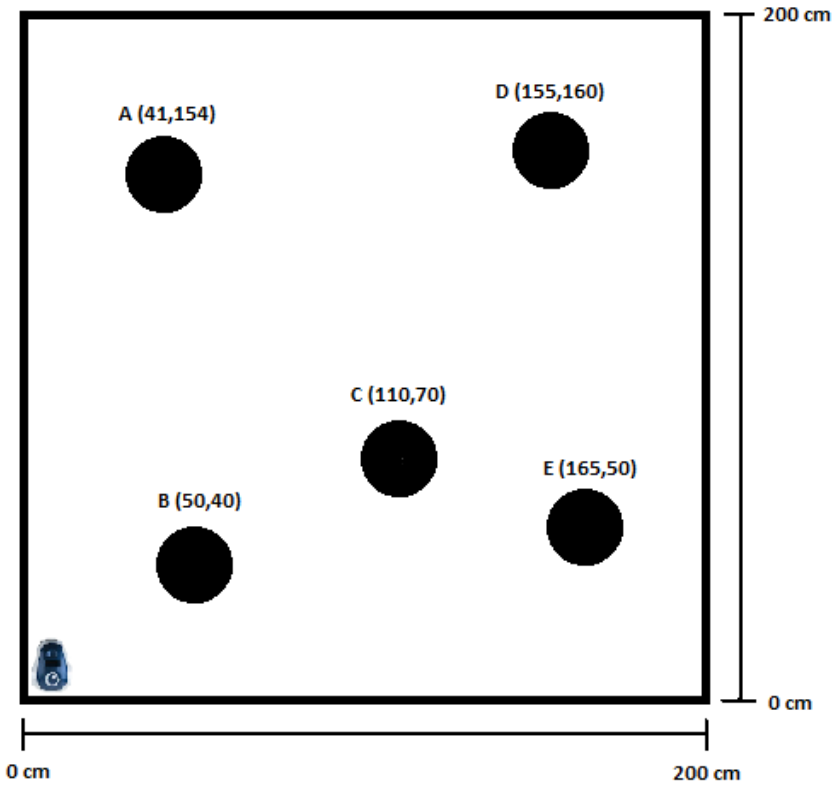
Nakon ovih koraka možemo prenijeti program na Moway platformu. MPLAB IDE će prilikom kompajliranja programa generirati datoteku s .hex ekstenzijom unutar direktorija u kojem se nalazi projekt. Ta datoteka prenosi se na platformu odabirom “Program .hex file“ naredbe u glavnom prozoru Moway World programa. Pritom Moway treba biti spojen na kompjuter USB kabelom. Nakon što se prenese .hex datoteka, može započeti testiranje programa u stvarnom svijetu.



Slika 26: uključivanje "linker" skripte u projekt

5. Testiranje zadatka

Testiranje je provedeno, kao što je već opisano, na parketu omeđen tamnoplavom izolir trakom koja označava granicu prostora pretraživanja površine $2 \times 2\ m^2$. Unutar tog prostora na pet različitih koordinata postavljen je crni krug promjera $20\ cm$ (slika 27), te je za svaki položaj provedeno deset pokusa za deterministička putanju i deset pokusa za stohastičku putanju. Mjerilo se vrijeme od početka gibanja Mowaya do pronalaska objekta pretraživanja.



Slika 27: Početni položaj Mowaya i položaji objekta kroz svih pet pretraživanja

Prilikom svakog pretraživanja Moway se nalazio u donjem lijevom kutu područja pretraživanja; to mu je bio početni položaj. Koordinate središta kruga odabrane su tako da za dvije koordinate, $A(41cm, 154cm)$ i $B(50cm, 40cm)$, Mowayu treba relativno malo vremena da nađe krug gibajući se determinističkom putanjom. Jedna koordinata, $C(110cm, 70cm)$ odabrana je tako da se nalazi blizu centra područja pretraživanja, a preostale dvije, $D(155cm, 160cm)$ i $E(165cm, 50cm)$, su odabrane tako da Moway pronađe krug nakon dugo vremena gibajući se determinističkom putanjom. Dok će

rezultati deterministićkih pretraživanja za pojedine položaje krugova biti slični, za stohastičko pretraživanje takve rezultate se može samo nagađati; zato je provedeno dovoljno eksperimenata kako bi se mogle usporediti efikasnosti oba načina pretraživanja.

U sljedećoj tablici napisana su vremena pretraživanja u minutama i sekundama za svaku koordinatu te za oba tipa pretraživanja, srednje vrijeme pretraživanja, te maksimalno odstupanje pojedinog vremena pretraživanja od srednjeg vremena pretraživanja σ_{max}.

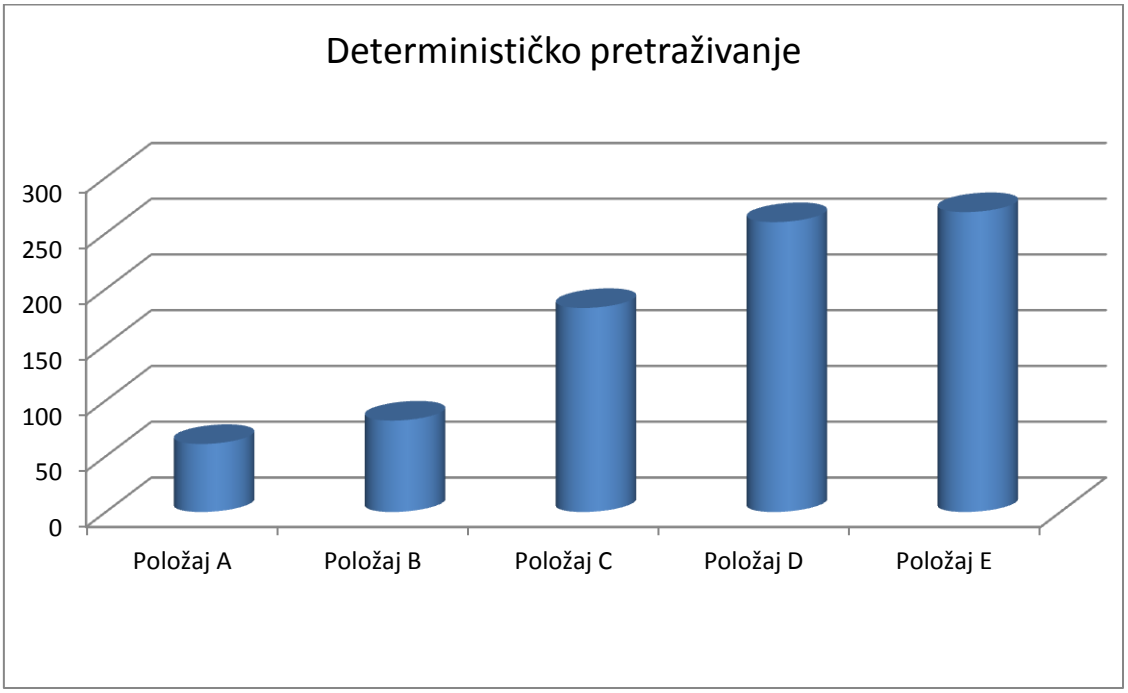
Koordinata i tip pretraživanja	Vrijeme pretraživanja u minutama i sekundama										Srednje vrijeme	σ_{max}
A, deterministićko	0:52	0:50	0:58	1:21	0:53	1:13	0:52	1:20	0:52	0:55	1:01	0:20
A, stohastičko	0:55	4:30	1:44	6:05	2:11	1:52	7:32	1:04	10:29	1:01	3:44	6:45
B, deterministićko	1:06	1:34	1:15	1:46	1:07	1:06	1:45	0:46	2:03	1:06	1:22	0:41
B, stohastičko	1:14	3:20	0:26	0:36	4:45	0:25	7:06	0:41	0:08	0:48	1:57	5:09
C, deterministićko	3:18	2:16	2:56	3:23	2:36	2:54	3:04	3:27	3:22	3:14	3:03	0:47
C, stohastičko	1:10	2:49	3:01	2:29	1:09	1:06	0:23	2:23	4:39	4:43	2:23	2:20
D, deterministićko	4:22	4:50	4:28	4:05	3:30	5:26	4:20	3:58	4:15	4:04	4:20	1:06
D, stohastičko	1:00	2:20	0:45	3:47	0:38	1:05	10:41	2:14	5:27	3:25	3:08	7:33
E, deterministićko	5:07	4:48	3:46	5:35	4:45	4:41	3:25	4:13	3:56	4:37	4:29	1:06
E, stohastičko	0:39	7:11	1:13	8:59	0:48	2:03	1:47	1:17	1:55	5:41	3:09	5:50

Tablica 1: vremena pretraživanja za svaku koordinatu te za oba tipa pretraživanja

5.1. Rezultati deterministićkog pretraživanja

Koordinate su imenovane prema abecednom poretku uzimajući u obzir srednje vrijeme deterministićkog pretraživanja: točka A biti će prva pronađena, a točka E zadnja. Rezultati deterministićkog pretraživanja su predvidljivi jer se zna koliki će otprilike put Moway prijeći prilikom svakog pretraživanja, pa se analogno tome zna i koliko će iznositi vrijeme pretraživanja. Također, vremena deterministićkog pretraživanja za isti položaj objekta su približno ista, te nema većih odstupanja od srednjeg vremena. Odstupanja se kod deterministićkog pretraživanja javljaju zbog nesavršenosti aktuatora, odnosno zbog toga

što se Moway neće gibati istom deterministčkom putanjom svaki put. Iznosi maksimalnog odstupanja od srednjeg vremena očekivano rastu s udaljenošću objekta od početnog položaja Mowaya: za položaj A maksimalno odstupanje iznosi 20 sekundi, za položaj B 41 sekundu, za položaj C 47 sekundi, a za položaj D i E je jednako i iznosi 66 sekundi. Odnos vremena determinističkog pretraživanja za različite položaje objekta prikazan je na grafu 1.

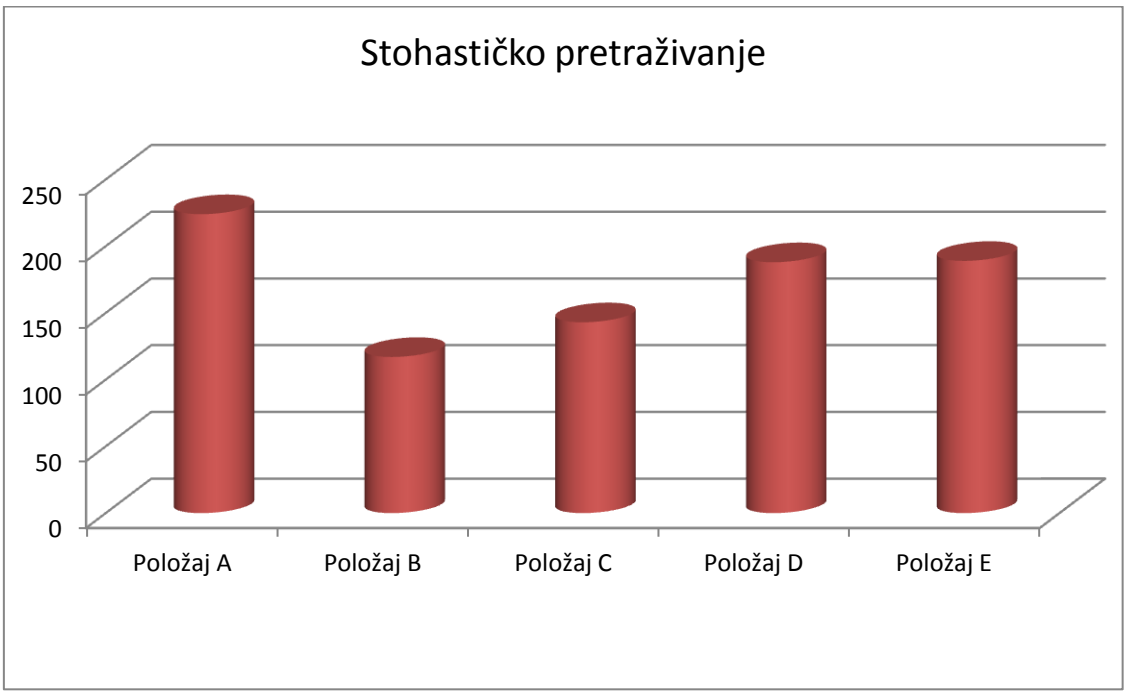


Graf 1: srednje vrijeme deterministčkog pretraživanja za različite položaje objekta

5.2. Rezultati stohastičkog pretraživanja

Za razliku od deterministčkog pretraživanja čiji su ishodi bili donekle predvidljivi, o ishodima stohastičkog pretraživanja ne može se govoriti s tolikom sigurnošću. Jedna od pretpostavki koje bi se mogle tvrditi jest da je srednje vrijeme stohastičkog pretraživanja manje za objekte koji su po zračnoj udaljenosti bliži početnom položaju pretraživanja: koordinata najbliža ishodištu jest koordinata B, zatim C, A, E i D. Srednja vremena ovih pretraživanja su: B(1:57), C(2:23), A(3:44), E(3:09) i D(3:08). Ovdje se vidi da je srednje vrijeme pretraživanja za bliže objekte manje nego ono za dalje objekte, iako vidimo da je za točku A, koja je srednja po udaljenosti, vrijeme pretraživanja najduže. Dobrog objašnjenja za ovu pojavu ne postoji, jer je pretraživanje stohastičko. Ukoliko se promotre rezultati pretraživanja u tablici 1, pod stohastičkim pretraživanjem A koordinate vide se

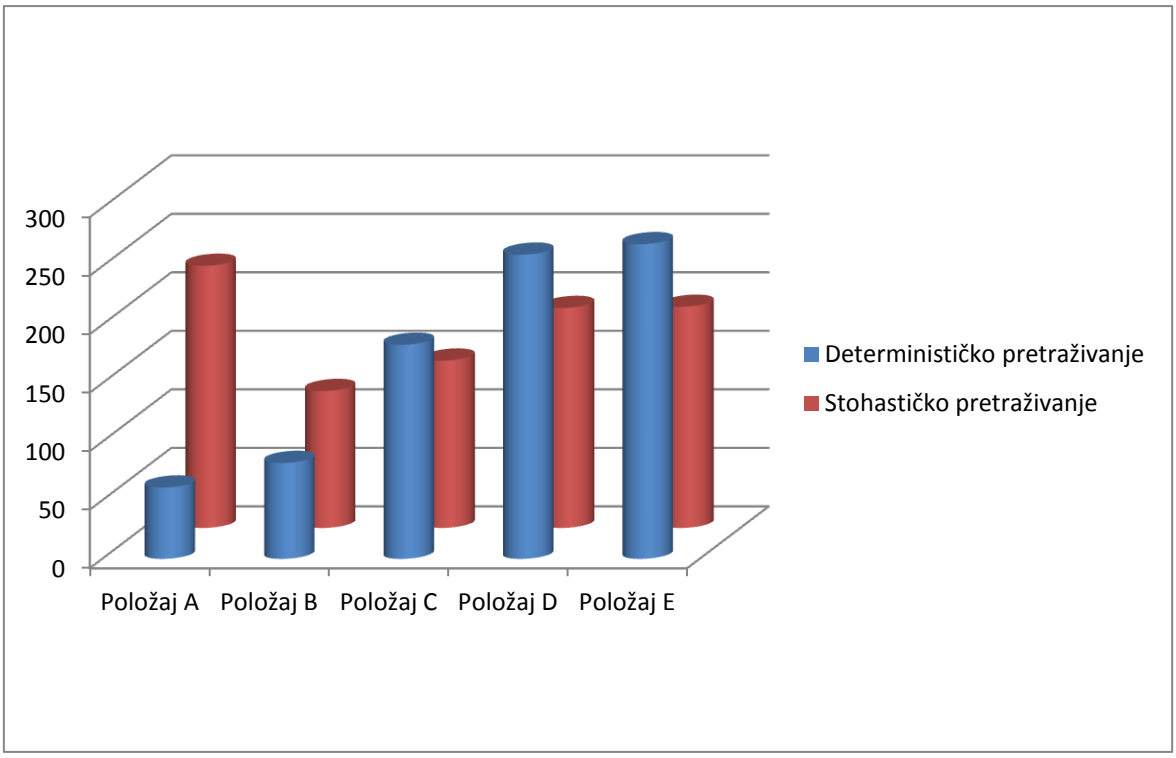
dva jako duga vremena pretraživanja, od 7:32 i 10:29 minuta, koji najviše utječu na povećanje iznosa srednjeg vremena. Takva vremena mogla su se naravno dogoditi i najbližoj i najdaljoj koordinati objekta pretraživanja. Zbog ovakvih dugih vremena pretraživanja maksimalno odstupanje stohastičkog pretraživanja od srednjeg vremena je puno veće nego kod deterministčkog (na primjer, 7:33 minuta). Odnos vremena stohastičkog pretraživanja za različite položaje objekta prikazan je na grafu 2.



Graf 2: srednje vrijeme stohastičkog pretraživanja za različite položaje objekta

Međutim, ukoliko se uspoređuju rezultati deterministčkog i stohastičkog pretraživanja, vidi se da se mijenja omjer srednjih vremena oba pretraživanja za položaje od A do E: za točke A i B koje će Moway deterministčkim putem brzo naći, stohastičkim putem treba duže vremena. No za položaje C, D i E, koji zahtijevaju duže deterministčko vrijeme pretrage, stohastičko pretraživanje trajati će znatno kraće, od 40 sekundi a počožaj C do 80 sekundi za položaj E. Stohastičko pretraživanje inače traje najkraće za položaj B, koji je najbliži početnom položaju prema zračnoj udaljenosti. Iz ovih razmatranja može se izvesti zaključak: ukoliko je objekt pretraživanja po zračnoj udaljenosti blizu početnom položaju, deterministčko pretraživanje trajati će kraće od stohastičkog, a ukoliko se taj objekt nalazi daleko u smislu puta kojeg će Moway prijeći tražeći ga deterministčkim načinom,

stohastičko pretraživanje u tom slučaju trajati će kraće. Usporedba ovih vremena prikazana je na grafikonu 3.



Grafikon 3: usporedba srednjih vremena stohastičkog i determinističkog pretraživanja za različite položaje objekta

Konačno, ukoliko izračunamo srednje vrijeme svih pedeset provedenih stohastičkih i determinističkih pretraživanja, dobit ćemo zanimljiv rezultat: srednje vrijeme determinističkog pretraživanja iznosi 171 sekundu, a stohastičkog pretraživanja 172 sekunde. Ovakav rezultat pokazuje da stohastički način pretraživanja ne zaostaje za determinističkim i da je definitivno vrijedan promatranja.

6. Zaključak

Cilj ovog rada bio je usporediti deterministički i stohastički način pretraživanja te njihovu efikasnost. Svakim danom autonomni sustavi postaju sve popularniji s ciljem olakšavanja raznih poslova, a među te poslove spada i zadatak pretraživanja prostora. Rezultati provedenih eksperimenata pokazali su kako stohastički algoritam pretraživanja nimalo ne zaostaje za determinističkim te da ga treba ozbiljno uzeti u obzir. Također valja napomenuti da je proveden samo jedan tip stohastičkog pretraživanja, Levyev let, te da postoje i napredniji tipovi stohastičkih algoritama. Uz brojne optimizacije i poboljšanja stohastičkog algoritma, može se s određenom sigurnošću govoriti da će stohastički algoritmi pretraživanja prostora u budućnosti postati efikasniji od determinističkih.

Za potrebe izvođenja eksperimenta u stvarnom svijetu korištene su dvije Moway platforme na koje su implementirani deterministički i stohastički algoritam pretraživanja. Moway roboti su se pokazali poprilično pouzdanima za ovakav tip zadatka te su prikazali umanjeni model pretraživanja prostora. Naravno da tu nije kraj i da se Moway roboti mogu koristiti za realiziranje raznih zadatka, te svojim ponašanjem u svojem svijetu malih dimenzija mogu poslužiti kao model za jedan veći, robusniji sustav.

Pretraživanje prostora korištenjem mobilne platforme Moway

U ovom radu uspoređeno je determinističko i stohastičko pretraživanje prostora. Programska rješenja za ova dva načina pretraživanja implementirana su na mobilnu platformu Moway. Kao primjer determinističkog pretraživanja korištena je oštrokutna zig-zag putanja, a kao primjer stohastičkog pretraživanja putanja poznata pod imenom Levyjev let, koja opisuje gibanje životinja u prirodi u slučaju nedostatka plijena. Algoritmi pretraživanja prostora testirani su putem Moway platforme nad područjem površine $2 \times 2\ m^2$, gdje je predmet pretraživanja bio crni krug promjera $20\ cm$. Provedeno je ukupno 100 testiranja, 50 testiranja determinističkog i 50 testiranja stohastičkog pretraživanja, te je pokazano kako stohastičko pretraživanje daje jednako dobre rezultate kao i determinističko.

Ključne riječi: autonomna mobilna platforma, determinističko pretraživanje, stohastičko pretraživanje, Brownovo gibanje, Levyjet let

Area searching by using a Moway mobile platform

In this work deterministic and stochastic area searching are compared. Program solutions for those two sorts of searching are implemented on a Moway mobile platform. An acute angular zig-zag trajectory is used as a pattern for deterministic searching, and as a pattern for stochastic searching a trajectory named “Levy flight“ is used. Levy flight pattern describes motion of animals in the nature when the food is scarce. Both algorithms for area searching are tested through Moway platform in the area of $2 \times 2\ m^2$ surface, where the object of searching was a black circle with $20\ cm$ diameter. In total, 100 examinations are made, 50 examinations for deterministic area searching and 50 examinations for stochastic area searching. Examinations showed that stochastic search gives results as good as the deterministic one.

Keywords: autonomous mobile platform, deterministic searching, stochastic searching, Brownian motion, Levy flight