SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNALSTVA**

ZAVRŠNI RAD br. 5045

**ESTIMACIJA RELATIVNE TRANSFORMACIJE IZMEĐU UPARENIH OČITANJA LASERSKOG SENZORA UDALJENOSTI**

Matija Vukić

Zagreb, lipanj 2017.



Sadržaj

[1. Uvod 2](#_Toc484018982)

[2. Lokalizacija robota 3](#_Toc484018983)

[3. Metode lokalizacije 4](#_Toc484018984)

[4. Odometrija 6](#_Toc484018985)

[5. Korelacija histograma 8](#_Toc484018986)

[6. Alati 10](#_Toc484018987)

[6.1. ROS – Robotski Operacijski Sustav 10](#_Toc484018988)

[6.2. Python 15](#_Toc484018989)

[6.3. Operacijski sustav Ubuntu 16](#_Toc484018990)

[7. Opis problema 16](#_Toc484018991)

[8. Opis okruženja i algoritma 18](#_Toc484018992)

[8.1. Opis mape, robota i poruka 18](#_Toc484018993)

[8.2. Opis algoritma 22](#_Toc484018994)

[9. Rubni slučajevi 35](#_Toc484018995)

[10. Zaključak 38](#_Toc484018996)

[11. Abstract 38](#_Toc484018997)

[Literatura 39](#_Toc484018998)

1. **Uvod**

Ovaj rad će upoznati čitatelja s osnovnim pojmovima u automatici i robotici koji su vezani za lokalizaciju robota u prostoru gdje će usredotočenost više biti na lokalizaciji u dvije dimenzije. Biti će opisan jedan od glavnih problema prilikom lokalizacije u nepoznatom prostoru, a to je određivanje točne relativne udaljenosti robota od nekog početnog polazišta. Čitatelj će se upoznati s nekima od osnovnih metoda kojima se rješava taj problem, ali će naglasak biti na metodi uparivanja uzastopnih laserskih očitanje te pronalaska relativnog pomaka te rotacije pomoću algoritma koji koristi kroskorelaciju histograma. Čitatelja će se također upoznati s izgledom i nekim alatima razvojnog okruženja ROS koje je korišteno za izvođenje simulacije kretanja robota. Algoritam korelacije dvaju uzastopnih očitanja lasera će biti pomno objašnjen uz prateće slike grafova. Opisane će biti i rubne situacije kada se može vidjeti utjecaj nedostatka pravokutnih objekata u prostoru na točnost odometrije robota.

1. **Lokalizacija robota**

Tek u zadnjih 20-ak godina eksponencijalnim napretkom tehnologija koje su pridonijele značajnom razvitku mobilne robotike , roboti se sve više koriste u jednostavnim ili repetitivnim zadacima koji su vrlo često automatizirani. Samim time što robotom ne upravlja čovjek moraju postojati načini da se robot snađe u prostoru bez prethodnog znanja izgleda prostora. Današnji roboti imaju u sebi računalo koje je uglavnom dovoljno moćno za izračune kompleksnijih algoritama u realnom vremenu. Roboti također imaju na sebi i više vrsta senzora (npr. laseri, sonari, infracrveni, toplinski, itd.) koji služe za prikupljanje podataka iz okoline. Kombinacijom računala i podataka iz senzora moguće je izvoditi algoritme i kalkulacije koje će opisati prostor samome robotu te će se tako robot moći snaći u nepoznatom prostoru čija se topografija može promijeniti.

Jedan od najvećih današnjih problema robota je istovremena lokalizacija i izgradnja karte nepoznatog prostora (eng. SLAM). Lokalizacija uključuje samo jedno pitanje a to je "Gdje je robot sada?", s time da moramo uzeti u obzir da se to pitanje odnosi gdje je robot sada u usporedbi s njegovim početnim položajem u prostoru. Na pitanje se odgovara na različit način ovisno u tipu robota te o tome da li se robot nalazi u zatvorenom prostoru ili u otvorenom prostoru. Odgovori na to pitanje su položaj i orijentaciju robota bilo to u polarnim koordinatama ili u kartezijskom koordinatnom sustavu ili ipak u geografskoj širini i dužini te pripadnoj orijentaciji.

1. **Metode lokalizacije**

Postoje mnoge metode za lokalizaciju robota ovisno o robotu, njegovim resursima i senzorima te o samoj točnosti tražene lokalizacije. Postoje dvije vrste lokalizacije:

* Lokalna - informacije se dobivaju od senzora na robotu
* Globalna - informacije se dobivaju od GPS-a i sl.

U sljedećem tekstu riječ značajka (eng. feature) podrazumijeva uočljiv atribut ili aspekt nekog predmeta.

Jedna od najjednostavnijih metoda je "Metoda najmanjih kvadrata" (eng. Least Squares Error) gdje se koristi metoda najmanjih kvadrata za regresijsku analizu podataka. Cilj te metode jest minimizacija pogreške gdje robot jest i gdje bi robot trebao biti tj. ona okvirno procjenjuje gradijent funkcije pomaka robota.

Praćenje pozicije (eng. Pose Tracking) metoda se koristi kada je poznata početna pozicija robota pa je potrebno samo pratiti njegovu poziciju kroz vrijeme. Metoda koristi ekstrakciju tj. izdvajanje značajki okoline koje se mogu uspoređivati te se tako kroz vrijeme može pratiti promjena položaja nekih uočljivih objekata.

Metoda višestrukih hipoteza (eng. Multiple Hypotesis Localization) pretpostavlja da početna pozicija nije poznata ali je poznata topografija mape. U ovome slučaju početnu poziciju može robotu pridodati korisnik ili robot uvijek može započeti iz iste pozicije. Ideja iza ove metode je da se detektira svojstvo te se preko njega stvaraju hipoteze o položaju robota naspram toga objekta kojemu pripada to svojstvo. Može se stvoriti nova hipoteza ili se može poboljšati neka od prethodnih hipoteza ili ipak eliminirati.

Monte Carlo metoda (skr. MCL) također poznata pod nazivom lokalizacija čestičnim filtrom. Provodi se estimacija pozicije i orijentacije robota kroz vrijeme kako se prati njegovo kretanje. Koristi čestični filter kao reprezentacija distribucije vjerojatnih stanja opisanih orijentacijom i pozicijom tj. svaka čestica predstavlja jednu hipotezu o stanju robota. Algoritam započinje s uniformnom raspodjelom stanja te se tijekom vremena ističu one hipoteze koje najvjerojatnije opisuju stanje robota tj. filtriraju se čestice.

Metoda iteracije najbližih točaka (eng. Iterative Closest Point) minimizira razliku između dvije skupine točaka tako da iterira između svake dvije točke te pronalazi onu kombinaciju koja daje najmanju grešku. Često se koristi pri rekonstrukciji 2D ili 3D površina nakon skeniranja. Tijekom izvođenja te metode jedna skupina točaka je fiksna tj. referentna dok se druga transformira tako da se najbolje slažu koordinatama u referentnom skupu. Postoje mnoge varijante ICP-a od kojih su point-to-point (usporedba točka-točka) , point-to-plane (usporedba točke-površina) i point-to-line (usporedba točka-linija) najpopularnije.

Metoda usporedbe očitanja (eng. Scan Matching) koristi dva uzastopna očitanja senzora robota poput lasera, sonara, … da se pronađe relativan pomak robota u prostoru. Razlike između dva očitanja senzora se mogu uočiti vrlo lako zbog učestalosti skeniranja tj. frekvencije dohvaćanja senzorskih podataka te o gustoći lasera kojih uglavnom ima od nekoliko stotina do nekoliko tisuća. Načina na koji se zapravo traže razlike između dva očitanja ima mnogo. Koriste se laseri (eng. Laser Range Finders) da bi vidio prepreke i odometrija kotača (eng. Wheel Odometry) da dobije okvirno stanje robota. Odometrija iz kotača ima određenu grešku zbog proklizavanja kotača ili nekog drugog razloga te se ona tada ispravlja pomoću izračunatih vrijednosti odometrije iz lasera. U ovom radu će biti opisan te implementiran način pronalaženja odometrije pomoću korelacije histograma podataka iz lasera.

1. **Odometrija**

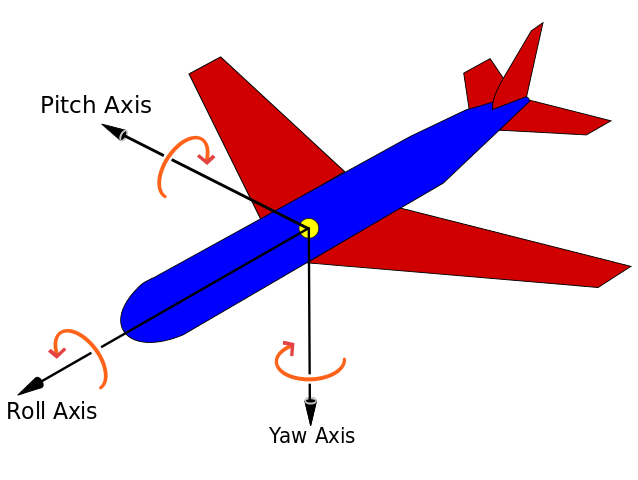
Riječ odometrija dolazi od grčkih riječi "odos" što znači "put" i "metron" što znači mjeriti. Odometrija obuhvaća određivanje lokacije naspram početnog položaja od kuda je robot krenuo. Odometrija se najčešće određuje preko računanja puta kojega su proputovali kotači u vremenu. Odometrija nam govori koordinate robota bilo to u 2D ili 3D sustavu te nam govori smjer robota tj. Kut rotacije robota naspram početne rotacije koja je uglavnom 0 stupnjeva tj. radijana. S obzirom da robot može biti rotiran oko x, y i z osi tada se rotacija daje u posebnom formatu.

Rotacija može biti prikazana kao rotacijska matrica veličine 3x3 te uglavnom prikazuje relativnu promjenu rotacije između dva koordinatna sustava **A** i **B**. Na sljedećim slikama možemo vidjeti pripadne rotacijske matrice za za svaku od koordinatnih osi.

Za razliku od rotacijskih matrica Eulerova reprezentacija rotacije se sastoji od samo 3 broja. Ta tri broja opisuju sekvencu kojom je objekt rotiran. Rotacija se može opisati na 12 načina zato što se prikazuje dinamička rotacija. Primjeri su XYX, ZZX, ZYZ, itd.

Rotacija se može također opisati i pomoću Roll-Pitch-Yaw (skr. RPY) načina. U ovome slučaju rotacije se vrše oko statičkih osi. Ovaj se način najbolje može opisati na primjeru rotacije aviona. Os skretanja (eng. yaw) je okomita na druge dvije osi i smjer joj je odozgo prema dolje.

Os poniranja (eng. pitch) je ona os koja se kreće od pilotovog lijevog krila prema desnom krilu. Os valjanja (eng. roll) je ona os koja se kreće od prednje strane prema zadnjoj tj. prolazi kroz cijeli avion po duljini. Te osi možemo mapirati na sljedeći način: skretanje – z os, poniranje – y os, valjanja – x os.



Slika 1 Model aviona s rotacijskim osima

Kao kompromis između prednosti i nedostataka prethodnih reprezentacija rotacije objekata koristi se sustav kvaterniona (eng. Quaternion). U matematici kvaterniona su skup brojeva koji proširuju kompleksne brojeve. Rotacija je prikazana kao vektor od 4 komponente. Na sljedećoj slici je prikazana temeljna formula za množenje kvaterniona.

Jedan kvaternion (jedinični) je definiran kao zbroj skalara i vektora . Iznosi:

.

Ako os rotacije opišemo u obliku gdje su , i  skalarne veličine te je kut zakreta te iste osi tada se rotacija može napisati kao:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Pretvorba kvaterna u rotacijsku matricu se provodi na sljedeći način:

1. **Korelacija histograma**

Metoda korištena za određivanje odometrije robota je Scan matching tj. uspoređivanje uzastopnih očitanja laserskog senzora. Algoritam koji će se zapravo koristiti za usporedbu podataka iz senzora se temelji na korelaciji histograma dobivenih nakon svakog očitanja. Histogrami općenito predstavljaju neku distribuciju numeričkih podataka. U ovom slučaju histogrami će predstavljati broj tj. frekvenciju laserskih zraka koje zadovoljavaju neku vrijednost.

U histogramu relativnog pomaka kuta na y osi će biti broj laserskih zraka između čijih je krajnjih točaka na objektu kut jednak kutu na x osi.

U histogramu relativnog pomaka udaljenosti će također na y osi biti broj laserskih zraka. Za dobivanje udaljenosti između dvije točke one će biti pretvorene iz polarnih koordinata u koordinate predstavljene x i y udaljenostima. Tako dobivenim točkama će se oduzetu x i y koordinate tj. dobit će se razlike te dvije točke. Te razlike ∆x i ∆y će biti na x osi histograma relativnog pomaka.

Ovim postupcima će se dobiti histogrami prethodnog i trenutnog očitanja preko kojih će se pomoću kroskorelacijske funkcije dobiti prave vrijednosti pomake u x i y smjeru te prava vrijednost rotacijskog kuta.

Da bismo usporedili dva očitanja moramo pronaći koliko su oni zapravo slični tj. različiti. U tu svrhu radimo njihovu kroskorelaciju. Kroskorelacija jest mjera kojom se prikazuje sličnost dvaju skupa podataka.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Funkcije i su diskretne funkcije. Koristiti će se diskretnu kroskorelacijsku funkciju. Ta funkcija je vrlo slična konvoluciji dvaju signala samo što se u slučaju konvolucije jedan signal mora zrcaliti u odnosu na y os. Ako uzmemo za primjer kroskorelaciju dvaju histograma promjene kuta, te na x os postavimo vrijednosti kuta, tada će na onoj lokaciji gdje je y maksimalan na ox osi biti ona vrijednost kuta za koju se najvjerojatnije razlikuju ta dva očitanja.

Mogu se kroskorelirati bilo koji nizovi podataka, ali to ne znači da su slični. Ta sličnost se najbolje može provjeriti normaliziranom kroskorelacijom. Normalizirana korelacija ima izlazne vrijednosti ograničene na vrijednost između -1 i 1 uključivo dok kroskorelacijska funkcija rješenja daje u proizvoljnom rasponu.

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Funkcija broj prikazuje normaliziranu kroskorelaciju gdje predstavlja tu finkciju, vrijednosti i predstavljaju srednje vrijednosti histograma h1 i h2. Naša očitanja su uzeta u istoj okolini u skoro istome vremenu pa će sličnost između dva očitanja biti prilično velika. To je i razlog zašto se ovakav algoritam koristi u Scan Match metodi. Sličnost se prikazuje najvećim brojem u grafu normalizirane kroskorelacije. Što je taj broj bliže jedinici to je veća sličnost dva očitanja.

Iz normalizirane kroskorelacije možemo dobiti i standardnu devijaciju tj. nesigurnost naše metode.

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Varijable je maksimalna apsolutna vrijednost normalizirane kroskorelacije, β je klasa histograma koja je u našem slučaju 1. Najmanja greška koju možemo dobiti je vrijednost klase histograma tj. jedan diskretni korak na x osi. Svi naši histogrami za svoje x osi koriste diskretnu podjelu vrijednosti. Ovaj algoritam neće uzimati u obzir vrijednosti normalizirane korelacija ili devijacija već će se samo gledati grubi podaci. X os za kutove će bit podijeljena po 1 stupanj od –90 stupnjeva do +90 stupnjeva. X od za udaljenosti će biti podijeljena po 0.05 m. Ti brojevi su dobiveni iz eksperimentnim testiranjem.

1. **Alati**
   1. **ROS – Robotski Operacijski Sustav**

Za potrebe implementacije i demonstracije koristit će se ROS. ROS je fleksibilna platforma za pisanje programa. To je skup alata, biblioteka i konvencija za pisanje programa koji će se izvoditi na robotu. ROS služi za apstrakciju robota tj. nije potrebno poznavati određenu konstrukciju nekog robota.

Slika 2 Logo aktualne verzije ROS-a

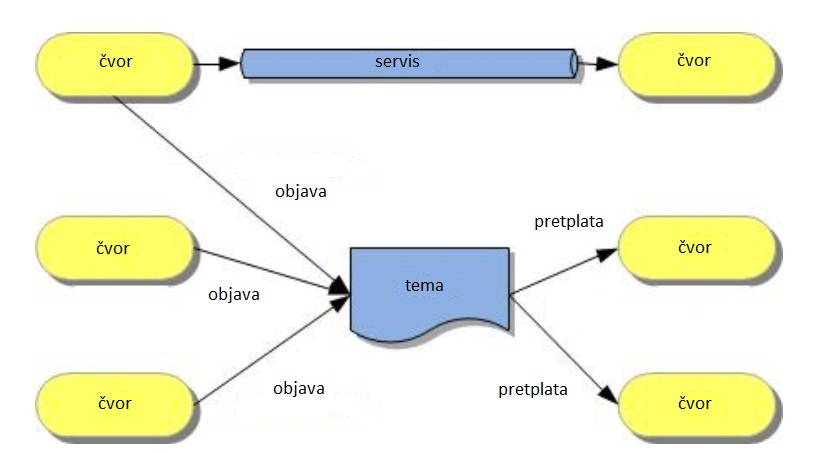
ROS sadrži razne vizualizacijske alate za prikazivanje robota u nekome prostoru. Cilj ROS-a je pojednostaviti proces stvaranja kompleksnih i robustnih ponašanja za više različitih robotskih platformi. Jedna od dobrih strana ROS-a je ponovna iskoristivost komponenata tj. jedan developer može napraviti neku funkcionalnost koju može konstruirati kao jedan čvor.

Prilikom izvođenja ROS je zapravo distribuirana mreža čvorova (eng. Nodes). Najvažniji dio je skup temeljnih čvorova potrebnih za pokretanje sustava. Taj minimalni skup se zove „roscore“.

Temeljni čvorovi su:

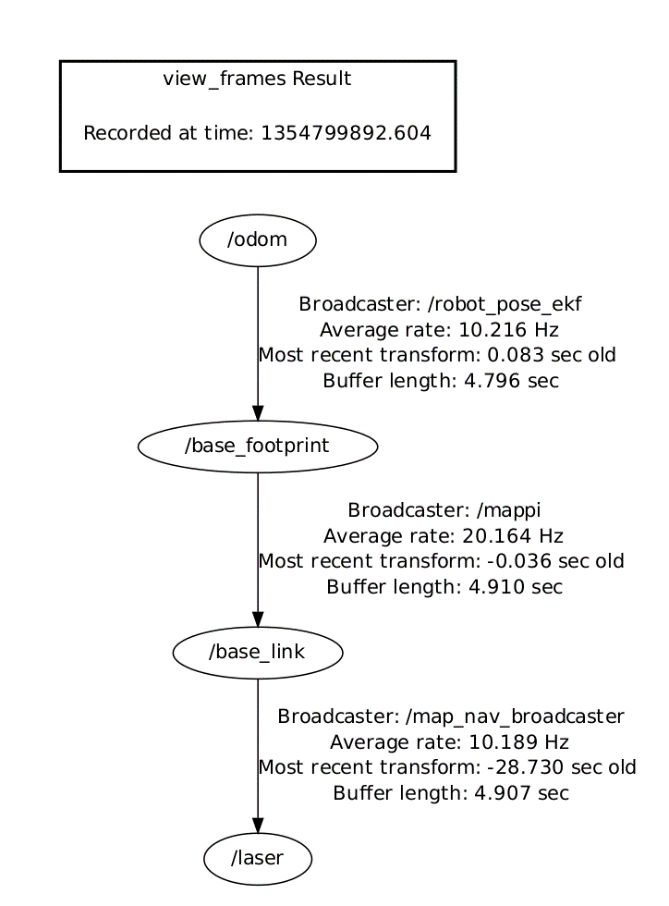
* Master – služi za registraciju i povezavanje čvorova, servisa te tema
* Parameter Server – rječnik globalno dostupnih parametara
* Logging čvor – čvor koji upravlja zapisnikom

ROS sadrži i teme (eng. topics). To su imenovane sabirnice preko kojih čvorovi izmjenjuju podatke. Čvorovi ne znaju s kime komuniciraju već to čine preko pretplatnika i izdavača. Teme su namijenjene za jednosmjeran tok podataka.

ROS sadrži i servise. Oni su najpogodniji za RPC zahtjeve i odgovore tj. za upite o nekoj temi ili čvoru. ROS pruža servise pod određenim imenom. Klijent pri pozivu servisa predaje mu upit na koji mu ROS vraća odgovor. 

Slika 3 Graf povezanosti komponenata ROS-a

Važan dio ROS-a su standardizirane poruke tj. strukture podataka koje se šalju između servisa i čvorova. Postoje razne unaprijed definirane poruke kao za odometriju (Odometry), orijentaciju (Orientation), brzinu (Twist) i mnoge druge. Moguće je konstruirati i posebne poruke koje trebaju korisniku ali se to ne preporuča.

Zbog toga što unutar ROS-a možemo imati više robota te svaki taj robot ima više senzora, svaki taj senzor ima svoj koordinatni sustav. Jedan koordinatni sustav se naziva okvirom (eng. frame). Svaki taj koordinatni sustav je na neki način transformacija nekog drugog koordinatnog sustava npr. u slučaju ovoga rada želimo prikazati pomak koordinatnog sustava robota s obzirom na koordinatni sustav cijele mape. U svrhu lakše transformacije koordinatnih sustava razvijen je paket **tf**. To je poseban paket koji korisnicima omogućuje praćenje više koordinatnih sustava unutar mape u realnom vremenu. Postoji i novija verzija **tf2** ali za potrebe rada je korištena starija verzija. Taj paket pruža metoda za dohvaćanje transformacija između neka dva koordinatna sustava (okvira). Ti koordinatni sustavi su spremljeni u obliku drveta koji na vrhu ima početni koordinatni sustav pa je tada svaka njegova grana onaj sustav koji se nalazi u njemu tj. prema kojemu postoji transformacija.

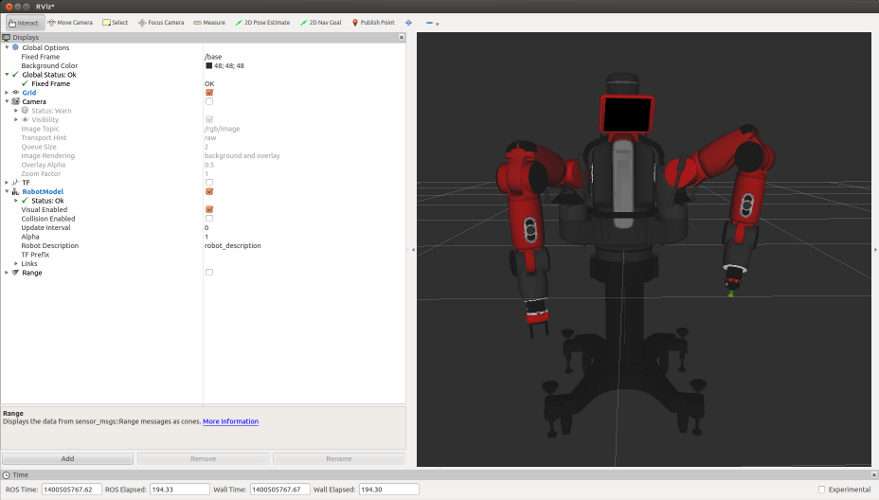
Slika 4 Struktura transformacija između okvira

Na slici broj 4 možemo vidjeti stablastu strukturu. Uz neke podatke može se vidjeti poveznica koja definira transformaciju između okvira „**/base\_link**“ i „**/laser**“. Ta transformacija postoji zato što su u stvarnom svijetu laseri zapravo pomaknuti od poda za neku udaljenost čime se definira okvir „**/laser**“. Okvir „**/base\_link**“ predstavlja koordinatni sustav robota kao cjeline te on uglavnom počinje od dna robota tj. od poda.

S obzirom na nezavisnost ROS razvojne okoline moguće ga je implementirati u bilo kojemu jeziku. Trenutno je sve funkcionalnosti ROS-a moguće koristiti u jezicima kao što su Python, C++ i Lisp, ali postoje eksperimenti s jezicima Java i Lua.

ROS razvojna okolina se trenutno izvodi na operativnim sustavima koji se temelje na Unix-u. Posljednja stabilna verzija ima kodni naziv „Lunar Loggerhead“ ali će implementacija ovoga rada biti izvršavana u verziji „Kinetic Kame“.

Jedan od važnijih alata za vizualizaciju podataka s robota alata unutar ROS razvojnog okruženja je Rviz. Na slici 5 može se vidjeti intuitivno korisničko sučelje.

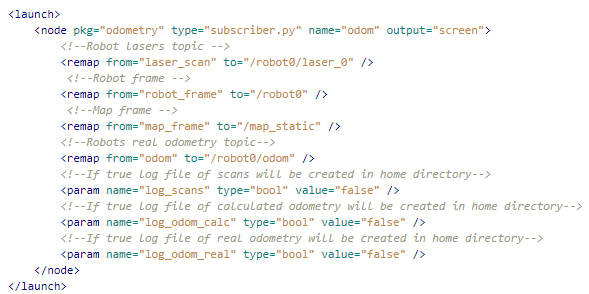


Slika 5 Prikaz sučelja alata Rviz

To je GUI program koji može vizualizirati ono što robot „vidi“ pomoću svojih kamera ili laser te također stanje robota tj. njegovu brzinu. Rviz će se tijekom ovoga rada ekstenzivno koristiti.

Za pokretanje već unaprijed definiranog robota koji već ima laserske senzore kao za već unaprijed definiranu mapu koristiti će se paket STDR simulator (eng. Simple Two Dimensional Robot Sumulator).

Još jedna specifična stvar za ROS je da nam omogućuje pokretanje složenijih kombinacija mapa i čvorova preko launch datoteka koje su tipa XML. Te datoteke mogu uključivati druge launch datoteke kao i čvorove.



Slika 6 Primjer sadržaja launch datoteke

Na slici 6 vidimo launch datoteku s nazivom našega čvora "odom" kao i s nazivima tema na koje će se pretplatiti naš čvor te još vidimo i zastavice koje govore čvoru dali da ispisuje log datoteke ili ne.

* 1. **Python**



Slika 7 Logo Python programskog jezika

Za potrebe ovoga projekta koristi se programski jezik Python , točnije koristi se verzija 3.5.2. Iako ROS ima biblioteke koje podržavaju verziju 2.7 one sasvim dobro funkcioniraju i s novijom verzijom. To je dinamički interpretirani jezik. Sintaksa jezika je prilično jednostavna te se u tu svrhu koristi često za izradu prototipova algoritama. Svi resursi iz ROS-a se mogu pristupiti preko sučelja koja su dostupna kao biblioteke za Python. Paket koji će biti korišten za crtanje raznih dijagrama je „matplotlib“. Za efikasnije izvođenje matematičkih operacije koristiti će se paket "numpy".

* 1. **Operacijski sustav Ubuntu**

ROS razvojno okruženje je podržano samo na operacijskim sustavima temeljenim na Unix-u. U tu svrhu Sve simulacije prikazane u ovome radu su bile izvršavane na operacijskom sustavu Ubuntu 16.04 LTS „Xenial Xerus“. Sve naredbe koje je potrebno izvršiti za pokretanje razvojnog okruženja su pokretane u terminal emulatoru.

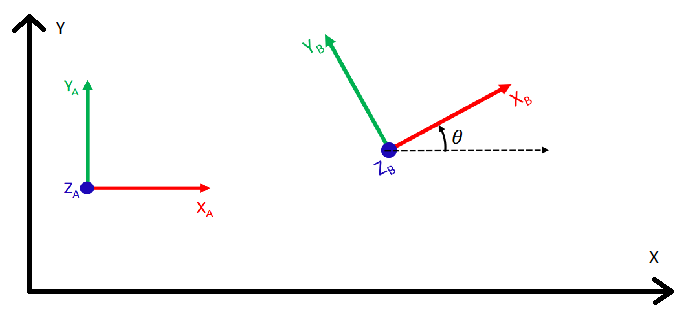


Slika 8 Logo operacijskog sustava Ubuntu

1. **Opis problema**

Kao što je opisano u opisu rada problem kojim se bavi ovaj rad je kalkulacija relativnog pomaka i relativne rotacije robota u nepoznatoj okolini. Zbog prirode izvora podataka uvijek će postojati neka greška pri svakoj usporedbi te će se ta greška akumulirati tijekom vremena. Biti će opisane osnovne formule korekciju greške kao i za pronalaženje standardne devijacije ali u okviru ove implementacije neće biti implementirano.

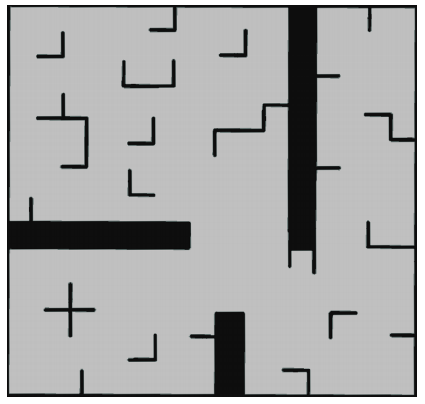
Potrebno je iz kolekcije laserskih očitanja, koji su definirani s polarnim koordinatama tj. s udaljenošću laserskog izvora od objekta i kutom pod kojim se ta laserska zraka nalazi ako bi se središte robota prikazalo kao središte kružnice, izračunati vrijednost kuta za koji se ta dva očitanja najbolje poklapaju. Nakon što se pronađe taj kut potrebno je zarotirati jedan skup očitanja za taj kut te tada pronaći i tj. pomak u smjeru x osi i pomak u smjeru y osi takve da se ta dva očitanja najbolje poklapaju po koordinatama. Taj će postupak trebati ponavljati između trenutnog očitanja te prethodnoga nakon čega će se kao prethodno očitanje postaviti trenutno, a kao trenutno će se koristiti novo očitanje sa senzora.



Slika 9 Promjena položaja i rotacije robota u globalnom koordinatnom sustavu

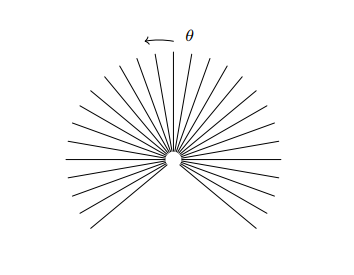
Opis problema može se sažeti u slici broj 9. Imamo globalni koordinatni sustav te prethodni položaj A i trenutni položaj B. Tražimo relativne pomake i tekut između položaja A i B**.**

1. **Opis okruženja i algoritma**

* 1. **Opis mape, robota i poruka**

Slika 10 Izgled standardne mape koja dolazi sa simulatorom

Rješenje će biti opisano na kao pronalazak pomaka robota kroz mapu na slici broj 10.



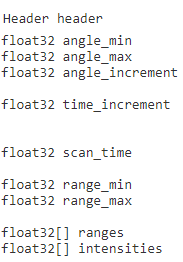
Slika 11 Distribucija lasera na robotu

Veličina mape je 15 metara u visini i 15 metara u širini, ali algoritmu nije to toliko bitno. Važno je napomenuti da će sve brojevne vrijednosti koje razne

teme objavljuju biti u SI sustavu jedinica. Samo će se nekada vrijednost kuta prikazivati u stupnjevima zbog lakšeg shvaćanja algoritma.

Stvarnog robota predstavlja apstraktni roboti koji ima dvije vrste senzora na sebi, a to su: sonari i laseri. U našemu slučaju samo će na biti potrebni laseri. Apstraktan opis robota može se vidjeti na slici broj 11.

Možemo vidjeti da robot ima velik broj laserskih zraka. Numeracija zraka počinje od desne strane ako robot gleda u istome smjeru kao i mi tj. suprotno od smjera kazaljke na satu. Pod numeraciju se smatra redni broj podatka o laserskoj zraci u nizu podataka koji se dobiju od teme (eng topic) na koju se naš čvor pretplati. Vrsta poruke koju tema koja odašilje podatke o laserskom očitanju je LaserScan. Struktura poruke se vidi na slici broj 12.



Slika 12 Struktura poruke LaserScan

Glavni atributi strukture su:

1. Header – zaglavlje s vremenom odašiljanja te okvirom na kojemu se odašilje
2. Minimalni i maksimalni kut – minimalni kut je kut na kojemu se nalazi prva laserska zraka, dok je maksimalni kut onaj na kojemu se nalazi zadnja laserska zraka
3. Diskretni korak inkrementacije kuta – to je onaj kut koji se nalazi između svake laserske zrake i izražen je u radijanima
4. Minimalna i maksimalna distanca – distance koje definiraju prihvatljive vrijednosti niza udaljenosti tj. distance laserskih zraka koje su ispod odnosno iznad tih vrijednosti se trebaju odbaciti
5. Niz udaljenosti – niz decimalnih brojeva koji predstavljaju udaljenost izvora laserske zrake i objekta

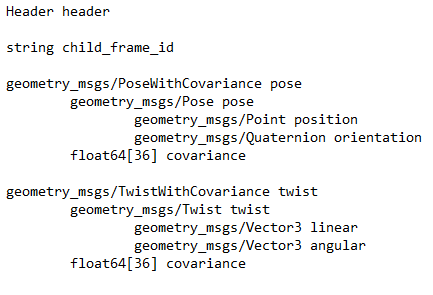
Vrijednosti koje su specifične za naš apstraktni robot su:

1. Minimalni kut
   * -120 stupnjeva ili oko –2 radijana
2. Maksimalni kut
   * 120 stupnjeva ili oko 2 radijana
3. Diskretni korak inkrementacija kuta
   * 0.06 radijana ili oko 0.36 stupnjeva
4. Minimalna udaljenost
   * 0.06 metara
5. Maksimalna distanca
   * 4.09 metara
6. Broj laserskih zraka
   * 667

Kut pod kojim se nalazi određena laserska zraka se računa na sljedeći način:

Kut = početni kut + diskretni korak inkrementacije kuta \* redni broj lasera

Još jedna važna struktura poruke koja će se koristiti je struktura koja predstavlja odometrijska stanja robota (eng. Odometry).



Slika 13 Struktura poruke Odometry

Struktura odometrijske poruke je:

Glavni atributi strukture su:

1. Header – zaglavlje s meta podacima
2. Child\_frame\_id – identifikacija okvira djeteta
3. Pose – opisuje statične stvari:
   1. Pose – pozicija prikazana pomoću x, y i z koordinata
   2. Orientation – orijentacija prikazana u obliku jediničnog kvaterna
4. Twist – opisuje dinamičke stvari:
   1. Linear – pokazuje linearnu brzinu kao vektor u 3 dimenzije
   2. Angular – prikazuje kružnu brzinu kao vektor u 3 dimenzije

Ova će metoda kao ulazne podatke dobivati poruku LaserScan, a kao rezultat će stvarati poruku Odometry.

* 1. **Opis algoritma**

Za pravilan rad algoritma potrebna su nam očitanja s laser koje dobivamo s teme '**/robot0/laser\_0**' i potreban je naziv okvira koji su specifični za robota '**/robot0**' kao i naziv okvira cijele mape '**/map\_static**' ili '**/map**' (ovisno o postojećim transformacijama).

Cijeli naš čvor **OdometryNode** je zapravo jedna klasa. Algoritam je zapravo jedna velika petlja zato što je metoda koja vrši kalkulacije postavljena kao povratna (eng. callback) metoda svaki puta kada se dobiju podaci. Simulator vraća podatke s frekvencijom 10 Hz. U početku čvor inicijalizira podatke specifične za izvor podataka tj. virtualni laser.

U sljedećim koracima oznaka S\_curr će biti istovjetna podacima trenutnog očitanja lasera, a S\_ref će biti istovjetno podacima prethodnog očitanja lasera.

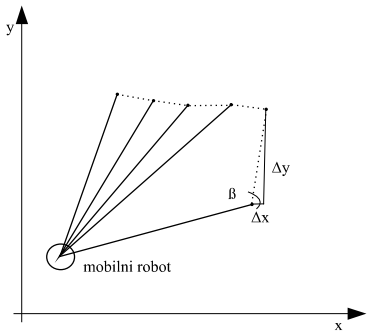
Koraci su:

* 1. Podaci za S\_ref očitanje

Dobivamo podatke o prvome očitanju koje tada filtriramo s obzirom na minimalne i maksimalno razlučljive udaljenosti lasera. Također računamo i kut pod kojim se nalazi ta laserska zraka kao i koordinate te točke u koordinatnom sustavu robota.

* 1. Računanje relativnih kuteva za S\_ref

Za stvaranje histograma kutova potrebno je prvo izračunate relativne kuteve između dvije točke. One se računaju na sljedeći način:



Slika 14 Primjer izračuna relativnog kuta između dvije točke

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

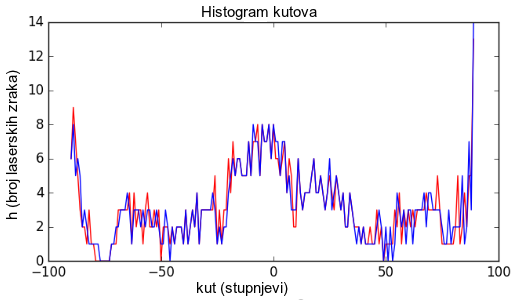
Zapravo gledamo koliki je broj točaka koje su pod nekim kutom kada uspoređujemo dvije susjedne točke.

* 1. Podaci za S\_curr očitanje

Dobivamo podatke o drugome očitanju te također filtriramo podatke i računamo potrebne vrijednosti.

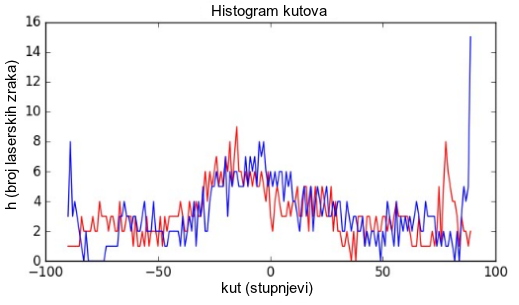
* 1. Računanje relativnih kutova za S\_curr

Računamo relativne kutove za drugo očitanje i stvaramo još jedan histogram.



Slika 15 Histogrami relativnih kutova bez rotacije

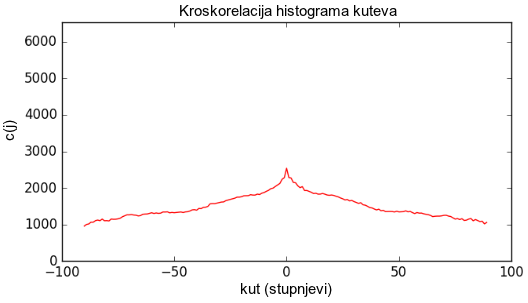
Na slici broj 15. je graf s oba histograma zbog boljeg uočavanja pomaka kuta ako ga je bilo. Crvene boje je histogram S\_ref, a plave boje je histogram S\_curr. Može se uočiti da se prilično dobro poklapaju, što znači da nije bilo zakreta.

Na slici broj 16. se uočava pomak između histograma što znači da je bio pomak u kutu tj. dogodila se rotacija.

Slika 16 Histogrami relativnih kutova prilikom rotacije

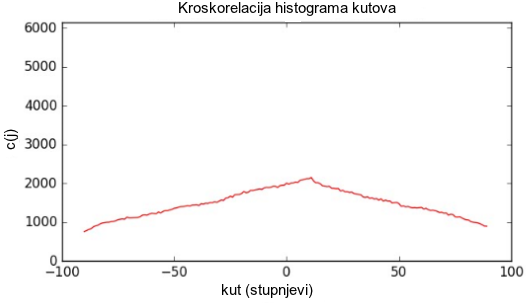
* 1. Kroskorelacija histograma kutova

U ovome koraku se vrši kroskorelacija ta dva histograma te se dobiva kut u stupnjevima za koji je najvjerojatnije drugo očitanje bilo zakrenuto naprema prvog ili 0 ako nije bilo zakreta. Rezultat je u stupnjevima.



Slika 17 Kroskorelacija histograma relativnih kutova bez rotacije

Na slici broj 17. se može primijetiti kako je maksimalna vrijednost grafa sa x koordinatom 0 što znači da nije bilo rotacije između dva očitanja.



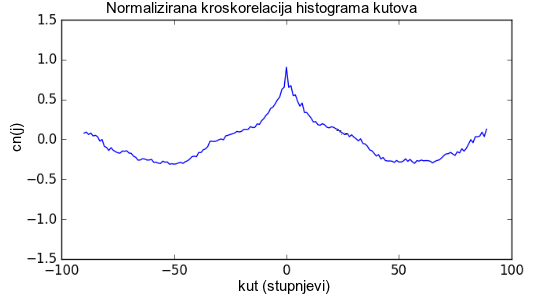
Slika 18 Kroskorelacija histograma relativnih kutova prilikom rotacije

Na slici broj 18. se može primijetiti da je maksimalna vrijednost grafa na nekoj x koordinati većoj od 0, što znači da se dogodila rotacija.

Vidi se da su vrijednosti na y osi vrlo velike što nam ne govori mnogo, samo da je ona vrijednost na x osi koja ima maksimalnu vrijednost na y osi pomak kuta.

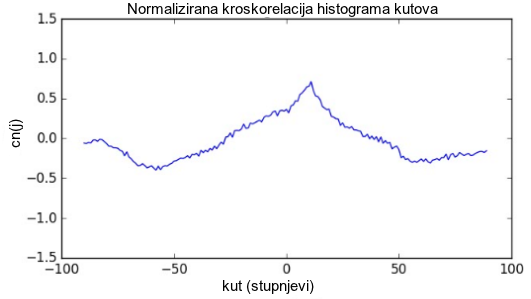
* 1. Normalizirana kroskorelacija histograma kutova

Ovo je opcionalan korak ali koristan je ako se želi vidjeti koliko dobro se slažu dva prethodna očitanja tj. da se prikaže kako se mogu korelirati dobro.



Slika 19 Normalizirana kroskorelacija histograma relativnih kutova kada nije bilo rotacije

Na slici broj 19. možemo vidjeti kako je također maksimalna vrijednost grafa na x koordinati s vrijednošću 0.



Slika 20 Normalizirana kroskorelacija histograma relativnih kutova prilikom rotacije

Na slici broj 20 se vidi kako je maksimalna vrijednost grafa na nekoj vrijednosti većoj od 0.

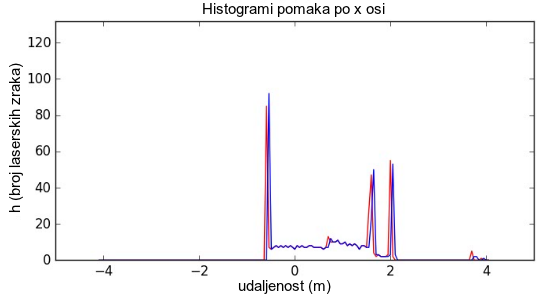
Vidi se da je maksimalna vrijednost na oba grafa približno 1 što je i očekivano.

* 1. Rotacija S\_ref tj. referentnog očitanja

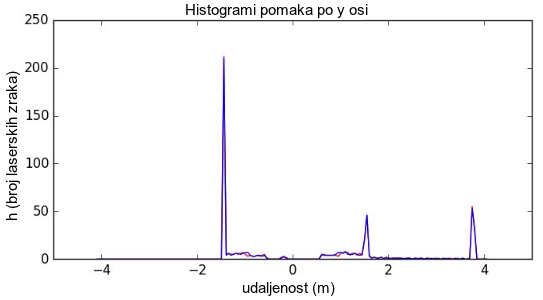
Sada kada imamo kut za koji je zakrenut S\_curr naprema S\_ref, rotiramo S\_ref za taj kut. Time dobivamo poklapanje očitanja u rotaciji.

* 1. Histogram pomaka po x i y osima za S\_ref i S\_curr

Sada možemo vidjeti koliko točaka u očitanju se nalazi na kojoj x koordinati. To činimo za oba očitanja.



Slika 21 Histogram pomaka po x osi

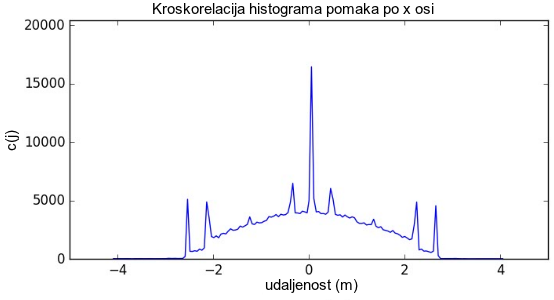


Slika 22 Histogram pomaka po y osi

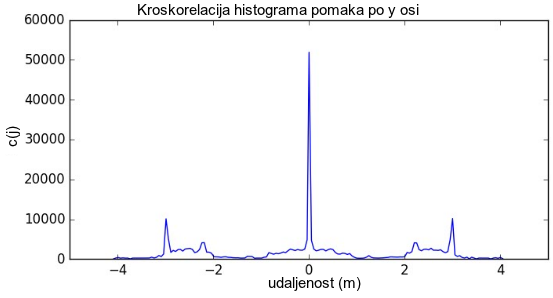
Na slici broj 21. se može vidjeti malo nepoklapanje te stoga znamo da se dogodio pomak. Dobili smo i pomak po y osi koji bi u idealnom slučaju trebao biti 0. Ovu vrijednost zapravo ne koristimo u računanju pomaka zato što je naš robot tipa robota koji se može samo kretati u x smjeru, a za kretanje u y smjeru mu je potrebna rotacija. Postoje roboti koje se mogu kretati u y smjeru bez potrebe za rotacijom

* 1. Kroskorelacija histograma pomaka po x i y osima

Sada preostaje napraviti kroskorelaciju histograma pomaka po x i y osima. Kao i u prošlome koraku dobit će se najvjerojatniji pomak po x i y osima, ali ovu vrijednost za y os neće se koristiti tj. ona bi trebala u idealnom slučaju biti 0.



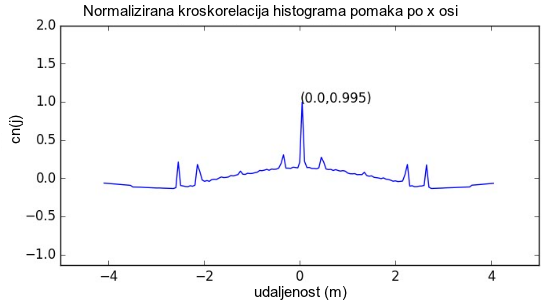
Slika 23 Kroskorelacija histograma pomaka po x osi



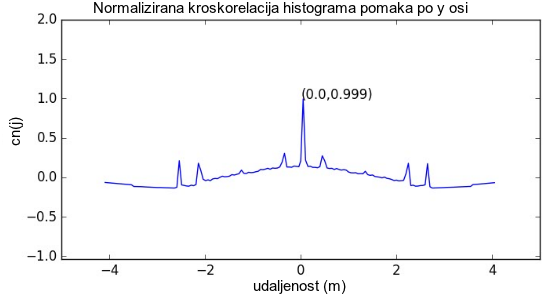
Slika 24 Kroskorelacija histograma pomaka po y osi

Na slici broj 23 vidimo kako je maksimalna vrijednost grafa pomaknuta u pozitivnom smjeru po x osi što znači da se desio pomak. Na slici 24 bi maksimalna vrijednost trebala imati x koordinatu 0 što u ovome slučaju i jest.

* 1. Normalizirana kroskorelacija histograma pomaka po x i y osima

Ovo je također opcionalan korak, ali služi da bi se vidjelo poklapanje očitanja.

Slika 25 Normalizirana kroskorelacija histograma pomaka po x osi



Slika 26 Normalizirana kroskorelacija histograma pomaka po y osi

Vidimo na slikama 25 i 26 kako su maksimalne vrijednost približno jedan što znači da su očitanja dobro korelirana.

* 1. Zamjena očitanja

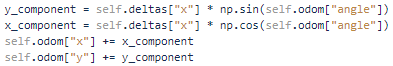
Zbog smanjenja posla koji se mora obaviti sada određene podatke o S\_curr spremamo u varijable S\_ref zato da ne moramo dva puta računati iste vrijednosti. Sljedeće očitanje će se spremiti u S\_curr , dok će prethodni S\_curr biti spremljen u S\_ref.

* 1. Objava odometrije

Sada kada imamo dobiven relativni kutni pomak i relativni translacijski pomak možemo objaviti novu lokaciju robota u koordinatnom sustavu mape. Moramo objaviti x, y koordinate i kut pod kojim se nalazi robot.

U prethodnim koracima kroskorelacija histograma kutova nam je dala pomak izmežu dva očitanja. Taj kut smo nadodali na globalnu vrijednost rotacije koja je u početku iznosila 0. Također smo kroskorelacijom histograma pomaka po x osi dobili pomak između dva očitanja.

Pomak u x i y koordinatamo pomoću pomaka po x osi dobivamo na sljedeći način:



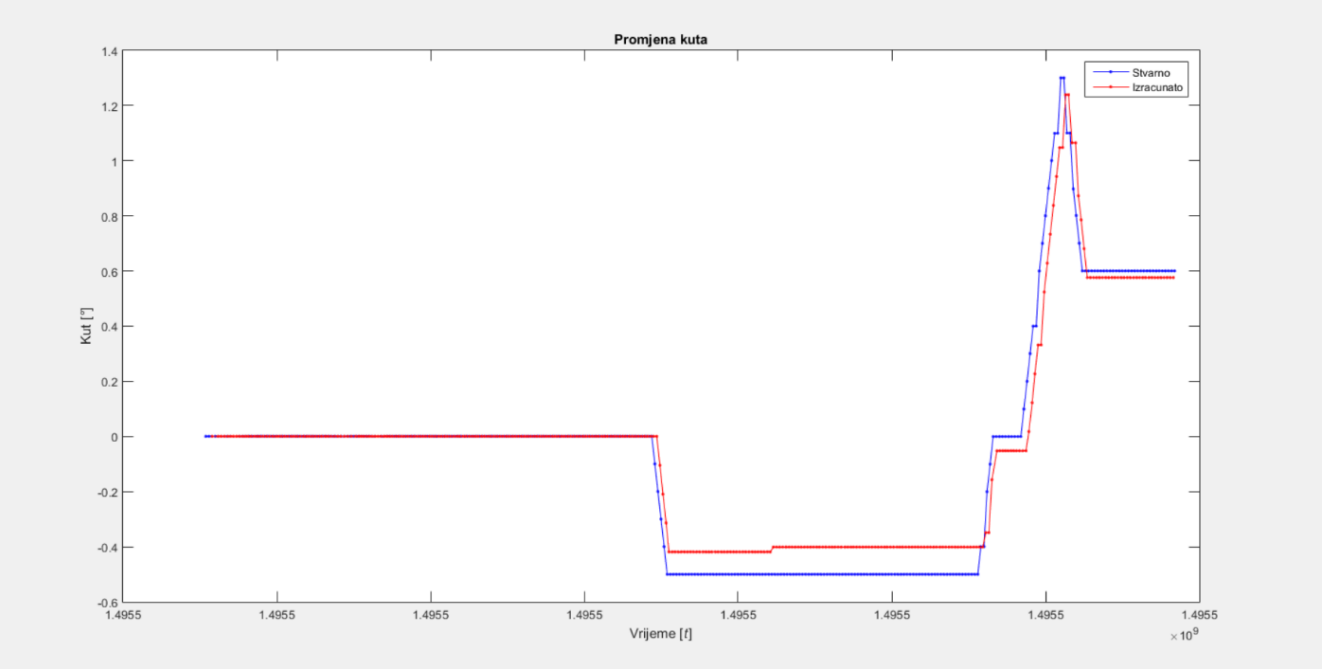
Slika 27 Dio koda koje računa sinusne i kosinusne komponente pomaka

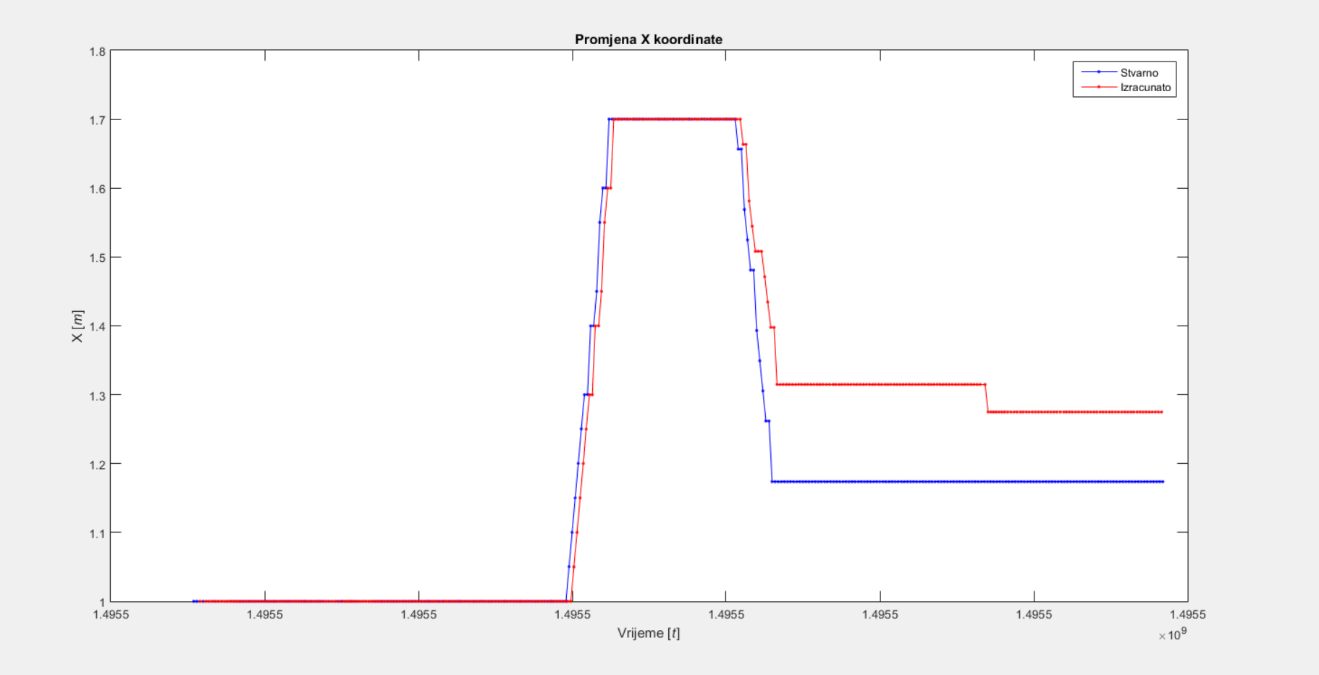
Varijable **y\_component** i **x\_component** su komponente pomaka, sadržanog u varijabli **self.deltas[„x“]**, na x i y osi globalnog koordinatnog sustava. Kada se dobiju te komponente one se pridodaju trenutnim koordinatama robota. Varijable **self.odom[„angle“]** sadrži trenutni kut rotacije robota.

Ako je robot pod kutem 0 stupnjeva tada će pomak po y osi biti 0, a pomak po x osi će biti jednak dobivenom pomaku. Ako je robot pod nekim kutom tada će pomak robota prikazan u koordinatnom sustavu mape biti dijagonalan. To znači da su pomaci po x i y osima zapravo sinusne i kosinusne komponente dobivenog pomaka.

* 1. Sada algoritam skače na korak 3. pa se sve ponavlja

Sljedeće slike su napravljene u programu MATLAB. Na njima se mogu vidjeti podaci koji su pročitani iz csv datoteke koje smo izveli iz našeg programa. Grafovi prikazuju usporedbu stvarne odometrije robota koju smo dobili pretplatom na temu '**/robot0/odom**' i odometrije koju smo mi izračunali.





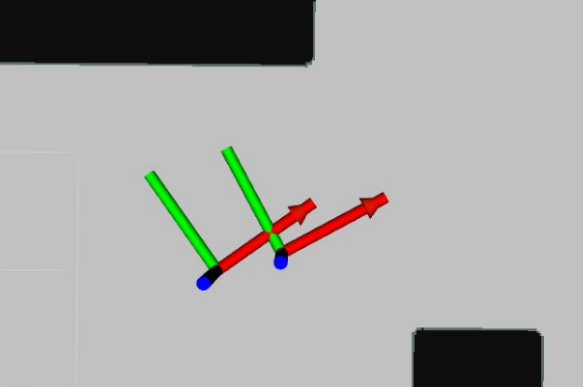
Slika 28 Usporedba stvarnog i izračunatog pomaka po x osi

Slika 29 Usporedba stvarne i izračunate promjene kuta

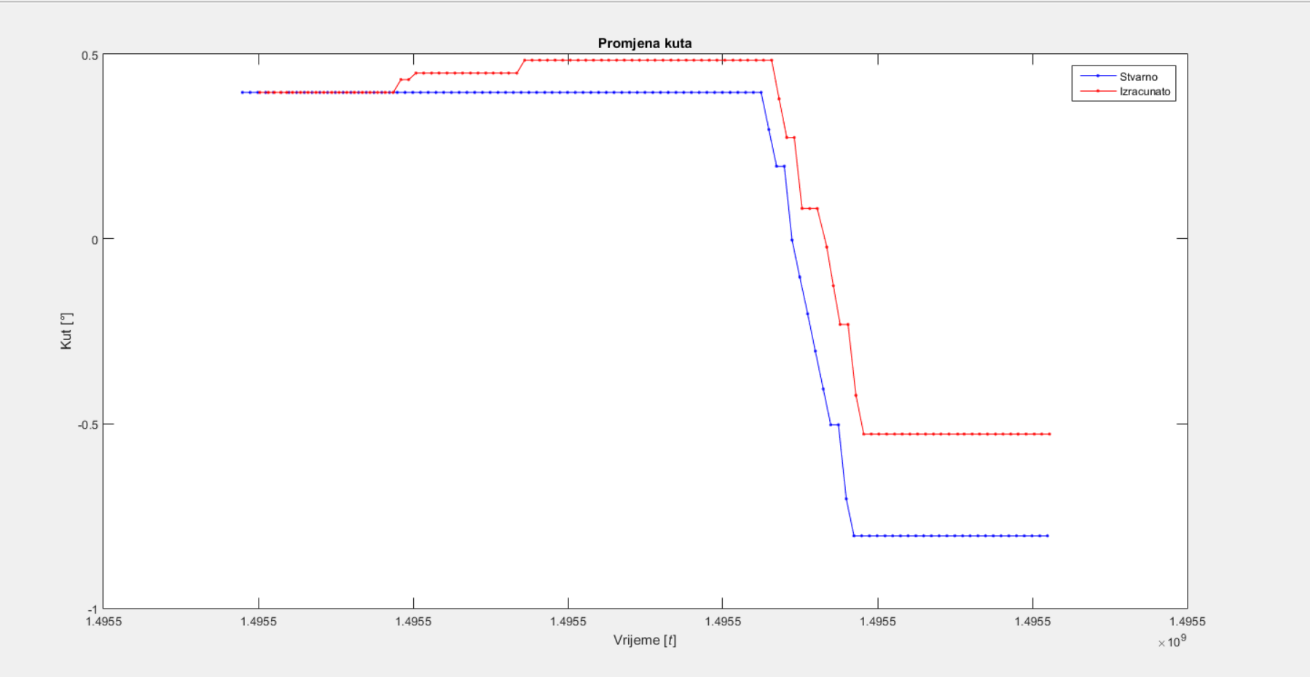
1. **Rubni slučajevi**

U ovome poglavlju će se vidjeti grafovi robnih slučajeva. Ovaj algoritam radi prilično dobro ako postoje mnogo pravokutnih rubova.

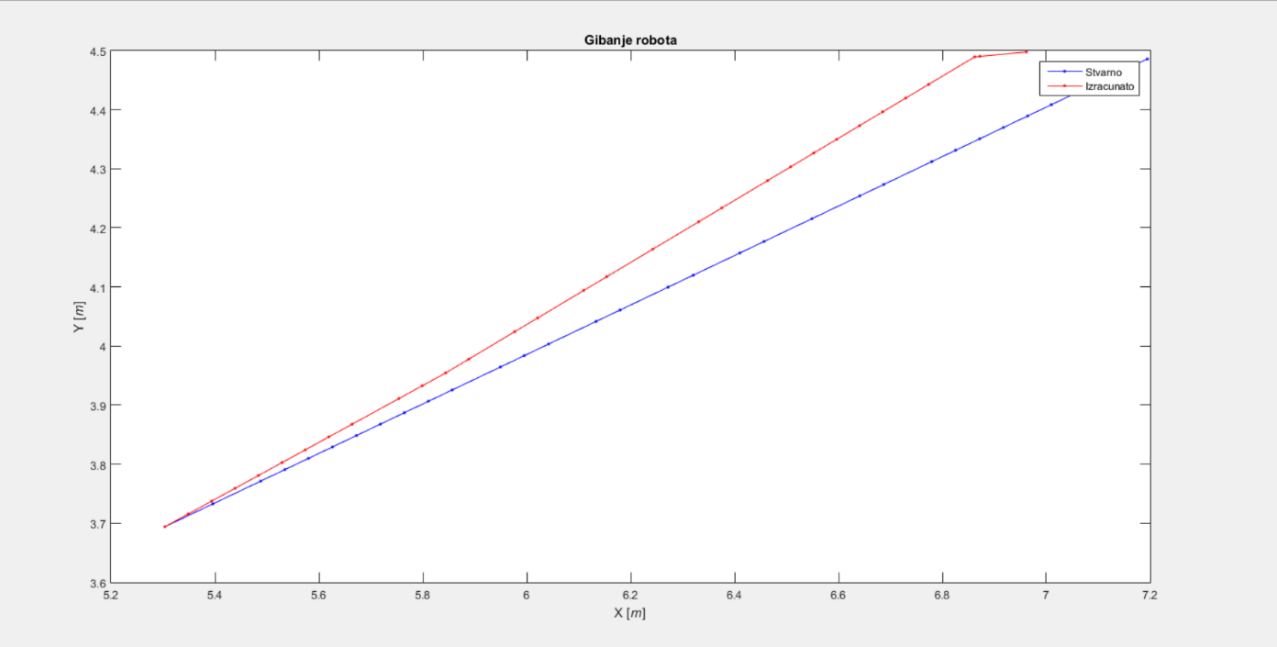
* 1. **Nedostatak rubova tj. pravokutnih objekata**

Ovdje vidimo kako nedostatak rubova utječe na računanje gibanja robota u prostoru kao i na računanje relativne promjene kuta kao i relativne promjene x i y pozicija.

Slika 30 Usporedba lokacije robota

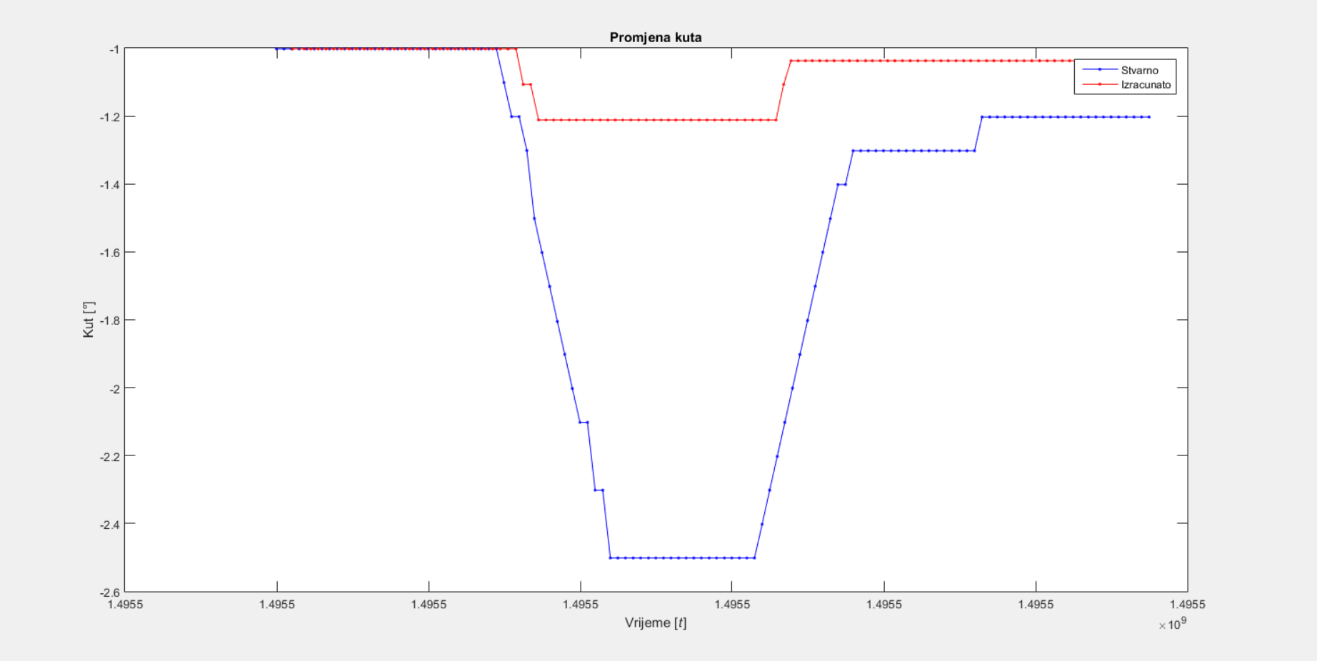


Slika 31 Usporedba promjene kuta s manjkom pravokutnih objekata



Slika 32 Usporedba x i y koordinata robota prilikom malog broja pravokutnih objekata

* 1. **Ravne površine**

Ovdje vidimo kako ravne površine utječe na računanje gibanja robota u prostoru kao i na računanje promjene kuta i relativne promjene x i y pozicija. Manjak pravokutnih zidova ovdje reprezentira samo jedan kut te dva okomita zida. Algoritam će detektirati promjenu samo u tome malome kutu što znači da neće imati dovoljno podataka za uspješno izvođenje pomaka.

Slika 33 Usporedba rotacije robota

Slika 34 Usporedba stvarne i izračunate promjene kuta

1. **Zaključak**

Vrlo je važno znati što točniji položaj robota u prostoru. Da bi se to saznalo postoje mnogi algoritmi zasnovani na različitim postupcima. U ovome radu je opisan i pokazana je implementacija estimacije lokacije pomoću kroskorelacije histograma. Taj algoritam nije savršen ali je prilično dobar u prostorima gdje ima pravokutnih objekata između kojih se mogu dobro vidjeti promjene kuta i pomaka. Algoritam bi se mogao još bolje optimizirati ako bi se ispravljale akumulirane greške što ova implementacija ne uzima u obzir. Mogao bi se također poboljšati ako bi se koristila već postojeća odometrija kotača te bi naš algoritam samo bio korišten za računanje i ispravljanje greške.

**Ključne riječi**: lokalizacija robot, odometrija, kroskorelacija, histogram

1. **Abstract**

It's very important to know the exact position of robot in space. To find out that position, there are many algorithms based on different procedures. This paper describes and demonstrates the implementation of location estimation by histogram cross correlation. This algorithm is not perfect, but is quite good in spaces where there are multiple rectangular objects between the changes of angle and shift can be seen clearly. The algorithm could be even better if the accumulated errors were corrected as this implementation did not take them into account. It could also be improved if the existing wheel odometry was used and our algorithm is used only for calculation and correcting of error.

**Key Words**: odometry, cross correlation, histogram,robot localization

# Literatura

**Edouard Ivanjko, Bojana Dalbelo-Bašić, Ivan Petrović. 2007.** *Correlation Based Approach to Mobile Pose Robot Tracking in Unknown Environments.* Zagreb : Faculty of Electrical Engineering and Computing, University of Zagreb, 2007.

**G. Weiss, E. v. Puttkamper.** *A map based on laserscans without geometric interpretation.* Kaiserslautern, Germany : University of Kaiserslautern, Department for Computer Science, Research Group v. Puttkamer.

**Gerhard Weiß, Christopher Wetzler, Ewald von Puttkamer.** *Keeping Track of Position and Orientation of Moving Indoor Systems by Correlation of Range-Finder Scans.* s.l. : University of Kaiserslautern.

**J. Borenstein, H. R. Everett , L. Feng. 1998.** *Where am I? Sensors and Methods for Mobile Robot Positioning.* Michigan : University of Michigan, 1998.

**Michael Bosse, Jonathan Roberts. 2007.** *Histogram Matching and Global Initialization for Laser-only SLAM in Large Unstructured Environments.* Kenmore, Australia : Autonomous Systems Laboratory, CSIRO ICT Centre, 2007.

**Myoung-Ho Kim, Min Kim, Kwae-Hi Lee. 2008.** *Localization Method Using Vector-Histogram.* Seoul, Korea : Department of Electrical Engineering, Sogang University,, 2008.

**Nuchter, Andreas.** *Multi-Robot Localization and Mapping based on Signed Distance Functions.* Germany : Julius-Maximilians-University Wurzburg.

**Röfer, Thomas. 2002.** *Using Histogram Correlation to Create Consistent Laser Scan Maps.* Bremen, Germany : Bremer Institut für Sichere Systeme, TZI, FB3, Universität Bremen, 2002.

Wikipedia. *Robot Operating System.* [Mrežno] https://en.wikipedia.org/wiki/Robot\_Operating\_System.

Wikipedia. *Odometry.* [Mrežno] https://en.wikipedia.org/wiki/Odometry.

Wikipedia. *Polar coordinate system.* [Mrežno] https://en.wikipedia.org/wiki/Polar\_coordinate\_system.

Wikipedia. *Ubuntu.* [Mrežno] https://en.wikipedia.org/wiki/Ubuntu\_(operating\_system).

Wikipedia. *Quaternions and spatial rotation.* [Mrežno] https://en.wikipedia.org/wiki/Quaternions\_and\_spatial\_rotation.

Wikipedia. *Quaternion.* [Mrežno] https://en.wikipedia.org/wiki/Quaternion.