SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI PROJEKT.

Egzaktna simulacija autonomnog istraživanja prostora pomoću višerobotskog sustava

Josip Spudić

Zagreb, srpanj 2022.

SADRŽAJ

1. Uvod

Istraživanje prostora pomoću robotskog sustava nalazi sve češću primjenu u robotici. Znanstvenici pokušavaju napraviti što bolji sustav za istraživanje prostora. Jedan od glavnih razloga uporabe robotskog sustava za istraživanje je vrijeme izvršavanja misije. Čovjeku bi za skeniranje i izradu karte prostora trebalo mnogo vremena i moguće je da napravi greške u skeniranju i izradi prostora. Nasuprot čovjeku, roboti su precizniji, mogu operirati u opasnim i nepristupačnim uvjetima, spremaju podatke koje kasnije lako prikazuju grafički. Neki od primjera korištenja robotskog sustava u kartirajnju nepoznatog prostoru su kartiranje prostora, pomoć unesrećenima u nepristupačnim prostorima, robotski usisavači, autonomna vozila i mnogi drugi.

Zbog sve većeg interesa u području autonomnog istraživanja, dolazi do potreba za različitim simulacijama na kojima se mogu testirati algoritmi. Egzaktna simulacija omogućuje istraživačima detaljno testiranje i usporedbu razvijenih algoritama uz adekvatno kartiranje prostora.

Egzaktna simulacija otkriva dvodimenzionalnu kartu bez korištenja algoritma istovremene lokalizacije SLAM[1]. Robotski sustav gradi kartu prostora na principu *fog of war* (otkriveni prostor postaje poznat i ne treba se opet otkrivati). Drugim riječima, simulirani robot postepeno otkriva prostor oko sebe uz savršenu informaciju o svojoj poziciji i orijentaciji, poznatoj karti i fronti istraživanja. Poznavajući sve te podatke jednostavno se implementira egzaktna simulacija robota u nepoznatom prostoru.

Ideja rada je napraviti egzaktnu simulaciju koja će poslije poslužiti za dodavanje različitih algoritama za autonomno upravljanje vozila, za bolje i brže otkrivanje prostora i korištenje višerobotskog sustava.

2. Autonomno istraživanje prostora

Ideja automatizacije sustava je minimizirati ljudsku potrebu za upravljanje uređajima. Automatizirani sustavi su precizniji, robusniji i brže donose odluke. U mnogim sustavima u kojima je implementirano autonomno upravljanje, pokazalo se kao pouzdano i efikasno rješenje za jednostavne zadatke.

Istraživanje prostora je vrlo kompleksna radnja. Za proces istraživanja prostora potrebno je navigirati robota, određivati prioritetne prostore istraživanja i brzo donositi odluke. Zato je još u mnogim istraživanjima glavni pilot robota čovjek. Čovjek dovoljno brzo i sigurno obavlja navedene zadatke pa nije potrebna automatizacija navedenih zadataka.

Zbog složenosti višerobotskog sustava, čovjek ne može upravljati takvim sustavom. Upravljanje višerobotskim sustavom zahtjeva brze odluke, mnogo izračuna i statistika. Zbog toga svi višerobotni sustavi su automatizirani. Autonomni sustav upravlja višerobotskim sustavom i brzo donosi odluke o putanji robota, izračunava optimalni put svih robota i usmjerava zadane robote u određene prostore. Automatizirani višerobotski (također i jednorobotni) sustav bi značajno skratio vrijeme istraživanja i time donio značajnu prednost izvršavanju različitih zadatka.

Automatizirani robot treba poznavati prostor oko sebe kako bi donosio odluke o sljedećoj akciji. Kako robot prepoznaje prostor oko sebe, on gradi kartu i pomoću nje se navigira i odlazi u prostore koje još nije istražio. Autonomno istraživanje bi zato trebalo imati algoritam koji će određivati planiranje puta kako bi što manje prošao kroz prostore koje je već istražio i s time uštedio vrijeme i električnu energiju u baterijama.

Radi jednostavnosti i fokusiranja na egzaktnu simulaciju, u radu je implementirano ručno upravljanje robotskim sustavom. Drugim riječima, autonomno istraživanje prostora u razvijenoj egzaktnoj simulaciji nije fokus ovog rada.

3. Metode i alati

Robotska aplikacija simulacije i sustav mapiranja implementirana je u *Robotskom operativnom sustavu (ROS)* [2]. *ROS* je korišten radi jednostavnosti izrade robotske simulacije i mnogih paketa koji nam omogućuju programiranje u C++-u ili Python-u. Za interferenciju s ROS-om u radu je korišten programski jezik Python. Python je odabran radi samointuitivnosti u programiranju i različitih paketa za obradu slika i strojno učenje. Za simulaciju napravljene su tri *Python* skripte. Prva skripta stvara odgovarajuću kartu prostora iz digitalne slike na kojoj je prikazana karta prostora. Druga skripta stvara robota, i robotske senzore i pozicionira robota u nepoznatu kartu. Treća skripta služi za kretanje robota upravljačkim naredbama s tipkovnice.

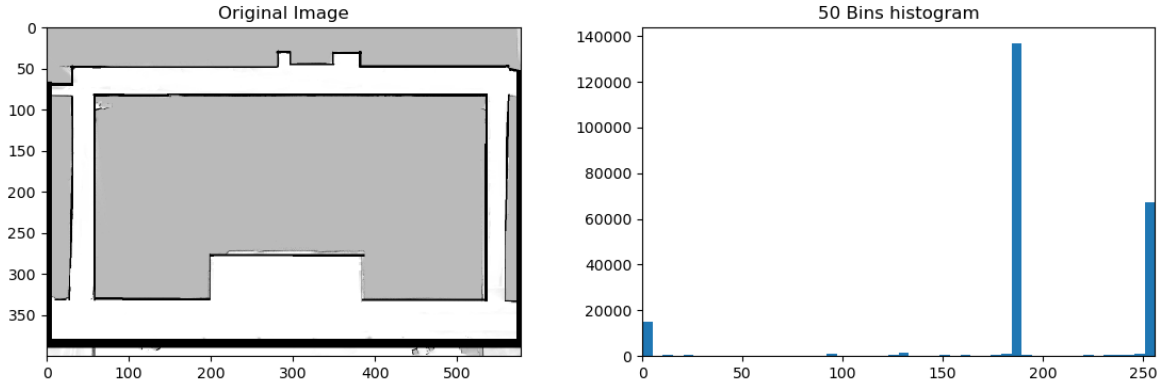
3.1. Karta prostora

Karta prostora je osnovna informacija koju će robot primati, ali i također koju će on slati. Karta prostora se sastoji od mreže ćelija. Mreža ćelija ima svoju visinu i širinu koja se iskazuje u broju ćelija. Ćelije predstavljaju zauzetost prostora. Prostor može biti slobodni, zauzet ili nepoznati prostor. Slobodni prostor je prostor gdje se robot, ali i ostali subjekti, mogu slobodno gibati. U slobodnom prostoru će biti stvoren robot koji će otkrivati druge ćelije prostora. Zauzeti prostor predstavlja zidove i druge objekte koji su prepeka za kretanje i senzor robota. Nepoznati prostor predstavlja neistraženi prostor za koji nedostaje informacija o zauzetosti. Karta prostora se stvara iz digitalne slike. Slika prikazuje kartu prostora s tri nijanse boje koje prikazuje tri različita prostora zauzetosti.

3.1.1. Slika prostora

Operacije nad slikom prostora izvršavaju se pomoću programskog paketa *OpenCV* [3]. Učitana slika se monokromira u sivu skalu koja poprima raspon vrijednosti od 0 za crni piksel i do 255 za bijeli piksel. Različite nijanse sive prikazuju različite vrijednosti zauzetosti. Svaka slika ima tri značajne nijanse sive prikazane slikom 1. Nakon što je slika učitana potrebno je odrediti granice prostora. Prostori se određuju ručno i najčešće bijele nijanse predstavljaju slobodni prostor dok crne zauzeti. Nijanse mogu biti različitno raspoređene

i nije uvijek da crna prikazuje zauzeti prostor, a bijela slobodni. Zato je potrebno odrediti granicu između prostora. Granica između prostora najlakše se definira iz histograma slike. Histogram grafički prikazuje distribuciju nijansi sive u slici.



Slika 1: Lijevo: slika karte, desno: histogram slike

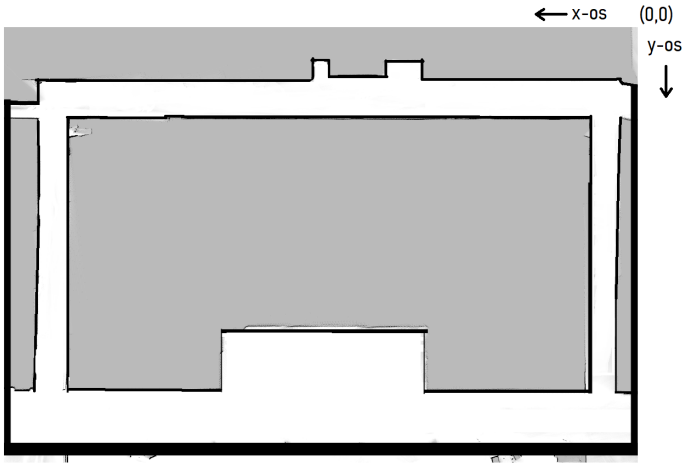
Na slici 1 prikazan je histogram s tri značajne vrijednosti. Prva vrijednost je na 0 i to predstavlja vrijednost zidova. Druga vrijednosti je oko 180 i prikazuje nepoznati prostor, a posljednja vrijednosti na 255 prikazuje slobodni prostor. Granica između nepoznatog i slobodnog prostora određena je iz histograma i definira se u yaml datoteci[4] spomenutoj u nastavku rada.

Veličina slike ima značajnu ulogu u kasnijoj brzini simulacije. Npr. ako je rezolucija slike postavljena na 836x652 (što je početna rezolucija slike na Slika 1 lijevo), vrijeme trajanja jednog simulacijskog ciklusa je viša od sekunde. Radi poboljšanja brzine simulacije potrebno je rezoluciju slike smanjiti. U radu su rezolucije smanjene za četiri puta, tako da je visina i širina slika prepolovljena. Nova reducirana karta ne gubi na vrijednostima prostora jer je prostor zadan konkretno i nema različitih konveksnosti i konkavnosti na zidovima. Ovim postupkom simulacija se značajno ubrza, a karta prostora ne gubi potrebnu informaciju o prostoru.

3.1.2. Mreža zauzeća prostora

Nakon učitanja digitalne slike prostora, slika se pretvara u format prilagođen komuniciranju u ROS-u . Tip poruke koji će predstavljati kartu prostora u ROS-u je mreža zauzeća prostora (*eng. occupancy grid*).

Mreža zauzeća prostora predstavlja dvodimenzionalnu mrežnu kartu, u kojoj svaka ćelija predstavlja vjerojatnost zauzetosti. Vjerojatnosti zauzetosti su vrijednosti između 0 i 100, gdje 100 predstavlja zauzeto, a 0 slobodno. Nepoznati prostor ima vrijednost -1. Također, mreža zauzeća prostora šalje osnovnu informaciju o mreži kao što su visina i širina mreže, ishodište mreže i rezoluciju.

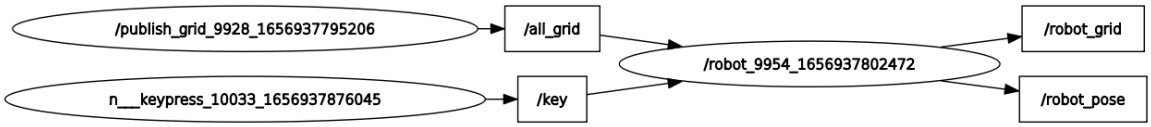


Slika 2: Mreža zauzeća

Slika 2 prikazuje mrežu zauzeća prostora. Na slici je prikazano ishodište karte u gornjem desnom kutu učitane slike kao i njezine smjerove x i y osi. Kasnije stvoreni robot u karti prostora će imati istu orijentaciju x i y osi kao i mreža zauzeća.

3.2. Robotski sustav

Robot je zamišljen kao mobilni robot koji istražuje prostor. Robot ima svoju poziciju i smjer gibanja. Robot je u ROS sustavu prikazan u obliku čvora koji je povezan s ostalim čvorovima radi izmjenjivanja informacije. Značajke robota su pozicija i orijentacija, njegova otkrivena karta prostora, kut vidnog polja i domet senzora.



Slika 3: Čvorovi i komunikacija robotskog sustava

Pozicija i orijentacija robota u ROS sustavu prezentirana je odometrijom (*eng. odometry*). Varijable odometrije koje robot koristi su pozicija u karti prostora (x,y) i orijentacija oko z-osi koja je prikazana u kvaternion sustavu.

Slika 3 prikazuje čvorove u ROS sustavu i komunikaciju između čvorova. Na lijevoj strani su dva čvora koja generiraju potrebne podatke za robota. Gornji čvor stvara poznatu kartu prostora iz digitalne slike i šalje robotu poruku u obliku mreže zauzeća(/allgrid). Doljni čvor prima ulaz s tipkovnice i robotu daje naredbe o kretanju s obzrirom na ulaz s tipkovnice. Srednji čvor je robotski sustav koji na ulazu prima navedene poruke. Nakon procesuiranja

tih poruka robot na svom izlazu objavljuje dvije vrste poruke. Prva poruka (/robotgrid) objavljuje otkrivenu robotsku mrežu zauzeća prostora. Druga poruka (/robotpose) objavljuje poziciju robota.

Otkrivenu kartu prostora robot izračunava svojim senzorima i prikazuje u obliku mreže zauzeća prostora. Robot konstantno kartira i nadodaje novi otkriveni prostor.

Kut vidnog polja i domet senzora opisuje značajke senzora koji robot ima na sebi. Kut vidnog polja je zadan u stupnjevima, a domet senzora je opisan veličinom koja predstavlja piksel.

Robot se u ručnom načinu rada giba pomoću strelica na tipkovnici. Različitim unosom robot mijenja svoju poziciju i orijentaciju. U trenutnoj implementaciji robot se može slobodno kretati u mreži zauzeća po svim vrstama ćelija. To znači da u radu nije napravljena limitacija za kretanje robota po zauzetom prostoru, tj. u simulaciji robot se može kretati kroz zidove.

3.3. Model senzora

Model senzora se nalazi na robotu i u simulaciji će imati istu orijentaciju i poziciju kao i robot. Robot u simulaciji konstantno skenira prostor oko sebe. Skenirani prostor interno sprema i objavljuje mrežu zauzeća prostora. Robotska mreža zauzeća prostora je na početku nepoznata i spremna da se popuni s otkrivenim prostorom (vrijednosti svih ćelija su -1).

Senzor ima svoje parametre. To su radijus ili domet senzora i kut vidnog polja. Domet senzora predstavlja koliko daleko senzorska zraka može otkrivati prostor od pozicije robota, a kut vidnog polja opisuje kutnu širinu senzora. U simulaciji domet senzora se zadaje u jedinci piksel, a kutevi se u simulaciji zadaju stupnjevima.

Robot skenira prostora u određenom radijusu oko sebe. Poznavajući poziciju robota i smjer orijentacije, računskim izračunom se određuje nalazi li se ćelija u mreži prostora u skeniranom prostoru senzora.

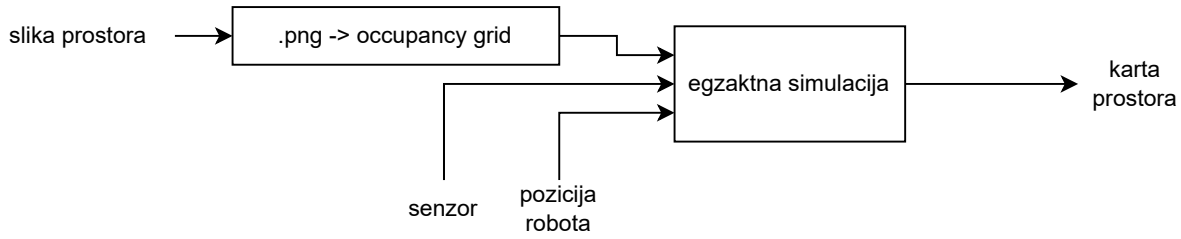
4. Egzaktna simulacija prostora

Za simulaciju i upravljanje robota najčešće se koristi simulator Stage. Stage simulator na jednostavan način simulira mjerenja i upravljačko sučelje kao prema stvarnim robotima[1]. Također, Stage koristi SLAM algoritam koji estimira poziciju robota i otkrivenog prostora. Kako bi se izbjegla estimacija i nesigurnost, napravljena je egzaktna simulacija.

Egzaktna simulacija prostora precizno izračunava kartu prostora robota. Poznavajući točnu poziciju robota i parametre senzora, robot izračunava prostor koji treba otkriti i uzima potrebnu informaciju iz slike prostora.

4.1. Dijagram sustava

Simulacija se može prikazati u pojednostavljenom blokovskom dijagramu. Slika 4 prikazuje grafički prikaz rada simulacije. Ulazi simulacije su slika prostora, senzor i pozicija robota. Ulazi se obrađuju i dobije se na izlazu otkrivena karta prostora.



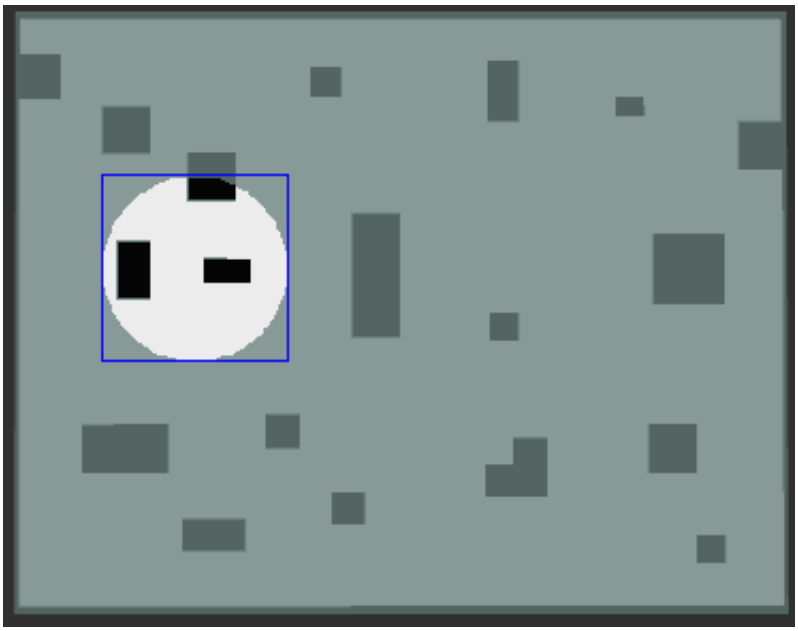
Slika 4: Dijagram egzaktne simulacije

Digitalna slika prostora se postupkom opisanom u poglavlju 3.1.1. pretvara u mrežu zauzeća prostora. Stvorena mreža zauzeća je potrebna egzaktnoj simulaciji kako bi simulacija znala koje su ćelije zauzete. Drugi ulaz robota je senzor. Senzor šalje radijus i kut vidnog polja senzora, a simulacija različitim algoritmima za senzor, poput emitiranja zrake ili kružnim senzorom, izračunava ćelije koje senzor zauzima s obzirom na poziciju robota. Senzor je samo otkrio prostor, ali ne zna informaciju o zauzetosti otkrivenih ćelije. Informacija o zauzetosti otkrivenih ćelija se dobi iz mreže zauzeća prostora. Te podatke precrtava u praznu kartu prostora i objavljuje ju u određenom kanalu.

4.1.1. Osvježavanje karte prostora pomoću kružnog modela senzora

U početku izrade simulacije za model senzora uzet je puni krug oko robota. Prostor koji će određivati senzor je krug u kojem je središtu robot. Senzor je određen radijusom udaljenosti robota i najdalje otkrivene ćelije. Kasnije se potvrdilo da je ovaj princip rada senzora mnogo sporiji. Također senzor izračunava vrijednosti iza i unutar zauzetih ćelija.

Poznavajući sliku prostora, poziciju robota i parametre senzora (kut je 360°, radijus se zadaje) iterira se kroz svaku ćeliju u karti prostora i ispituje je li udaljenost te ćelije manja od radijusa. Ako je udaljenost ćelije koja se ispituje manja od zadanog radijusa, na novoj karti prostora (/robotgrid) se preslikava ta ćelija.



Slika 5: Primjer kvadrata oko prostora istraživanja

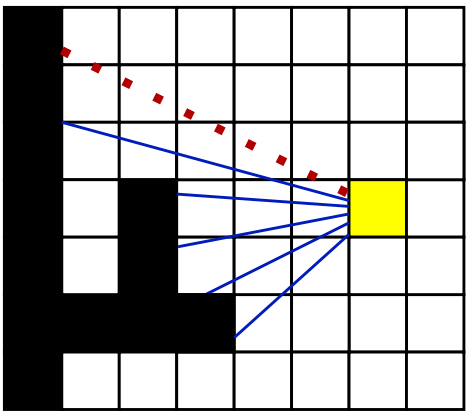
Opisani način rada je dugotrajan i zadaje probleme simulaciji. Za poboljšanje vremenskog ciklusa senzora smanjena je mreža zauzeća prostora koja se ispituje. Umjesto iteriranja kroz cijelu kartu prostora, iterirati će se u određenom kvadratu gdje je sigurno da će se kružnica radijusa nalaziti. Na slici 5 prikazan je primjer kvadrata oko kružnice senzora.

Svejedno i s optimizacijom, navedeni način rada nije davao željene rezultate jer je i dalje bio prespor za simulaciju (otprilike duplo sporiji od sljedeće metode). Ali najveći problem ovog tipa senzora je što otkriva i prostor iza zidova koji realni senzor ne može otkriti. Radi toga je odbačen kružni model senzora i odabran novi način rada senzora.

4.1.2. Emitiranje zraka

Emitiranje zraka (*eng. raycasting*) [5] bi riješio problem otkrivanja prostora iza zida koji je bio prisutan kod otkrivanja u kružnom modelu senzora. Robot emitira konačno mnogo

zraka i ako zraka dođe do prepreke ili prijeđe domet senzora, prebacuje se na sljedeću zraku.



Slika 6: Skica emitiranja zraka

Na slici 6 prikazan je primjer emitiranja zraka. Robot (označen žutom ćelijom) emitira zrake (označene plavom linijom) oko svog vidnog polja. Emitira se prvo (donja) zraka i kada zraka detektira zid, ili izgubi svoj domet, prebacuje se na sljedeću zraku zakrenutom za određeni kut. Kut zakreta je kut između dviju uzastopnih zraka. Kut zakreta ne smije biti velik da ne bi došlo do gubitaka ćelija. Primjeri gubitaka ćelija zbog velikog kuta zakreta opisan je u poglavlju 6.2.

Zraka putuje u prostoru diskretnim koracima. Crvena linija je primjer kretanja zrake. Zraka radi korake i ispituje je li je ta ćelija zauzeti prostor. U početku ispucana zraka je duljine jednog koraka i povećava se za korak dok ne dođe do zauzetog prostora (ili dok duljina zrake ne prijeđe duljinu radijusa). Korak zrake prikazan je razmakom između dvije crvene točke. Zbog manjeg koraka moguće je da zraka ispita ćeliju više puta. Problem redundancije ispitivanja ćelije je moguće riješiti DDA algoritmom [5], ali navedeni algoritam nije implementiran u radu. Korak zrake u radu je zadan na duljinu jedne ćelije. Korak zrake nije moguće mijenjati već će se mijenjati kut zakreta senzora kako bi senzor otkrio potrebne ćelije u svojoj okolini.

4.2. Yaml datoteka

Za prijenos i pohranu podataka u ROS sustavu koriste se yaml datoteke. U radu se yaml datoteke koriste za učitavanje različitih parametra za simulaciju. Potreba za yaml datotekama je došla zato što se u radu eksperimentiralo s različitim slikama prostora, različitim senzorima i početnim pozicijama robota. Yaml datoteke se pokreću prije pokretanja simulacije i vrijednosti yaml datoteka se spremaju kao ROS parametri.

Listing 4.1: Yaml file

```
image: /home/josip/ZavrsniRad/maps/2.png
position: [250, 340, 0.0]
height: 400
width: 578
boundary: 220
radius: 30
fov: 60
step: 1
```

Početni parametar (*image*) je slika prostora u kojoj će se odvijati simulacija. Slika prostora je zadana kao putanja u direktoriju. Parametar *position* predstavlja poziciju robota: x, y i z os. *height i width* predstavljaju visinu i širinu ćelija od mreže zauzeća prostora. *height i width* nisu stvarna rezolucija digitalne slike prostora već njezine umanjene vrijednosti (u radu visina i širina su se smanjivale za duplo). Nad slikom se obavlja smanjenje rezolucije kako bi se ubrzao program. Granica (*boundary*) se određuje iščitavanjem iz histograma, a zatim upisivanjem vrijednosti u .yaml datoteku. Vrijednost *boundary* parametra dijeli nepoznati prostor od slobodnog prostora. *radius* predstavlja doseg zraka koje koristimo u *raycasting-u*. *step* opisuje veličinu koraka kada se robot giba pomoću strelica s tipkovnice. Jedan *step* ima veličinu jedne ćelije.

4.2.1. Upravljanje robotskim sustavom

U završnom radu napraljeno je ručno upravljanje robotom. Korisnik preko tipkovnice strelicima zadaje orijentaciju i pomak robota. Strelica gore (↑) predstavlja pomak unaprijed i veličine je jedne ćelije. Suprotno, strelica dolje (↓) predstavlja pomak robota u suprotnom smjeru od njegovog usmjerenja. Veličina pomaka robota može se mijanjati yaml parametrom *step*. Strelica lijevo (←) pozitivno zakreće robota oko z-osi, a strelica desno (→) negativno zakreće robota oko z-osi. Veličina zakreta robota u lijevo i desno je veličine jednog stupnja.

5. Simulacija s višerobotskim sustavom

Opisanu simulaciju potrebno je proširiti na višerobotski sustav. Višerobotnim sustavom bi se ubrzalo kartiranje prostora i time smanjilo vrijeme istraživanja. Vrijeme je ključna komponenta većine primjena istraživanja nepoznatog prostora (npr. traženje unesrećenih osoba). Zato su mnoga istraživanja u robotici okrenuta različitim načinima ponašanja višerobotskih sustava i njihovoj primjeni u stvarnom svijetu.

U radu je implementiran ručno upravljiv višerobotski sustav. Svaki robot u višerobotskom sustavu ima značajke kao i u simulaciji s jednim robotom. Pojedinačni robot gradi svoju kartu prostora bez poznavanja karti prostora drugih robota. Zajedničkom komunikacijom te se karte prostora generiraju u jednu globalnu kartu prostora sa svim evidentiranim otkrivenim prostorom.

Slika 7: Primjer višerobotnog sustava

Ideja je u budućnosti napraviti automatski način rada višerobotnog sustava. Različitim algoritmima višerobotski sustav bi određivao prostore koje treba prvo otkriti i koliko robota treba poslati u te prostore.

6. Simulacijski rezultati

Nakon napravljene simulacije, provedena su različita istraživanja brzine i rada simulacije. Simulacija se prikazuje pomoću alata *RViz* [6]. RViz grafički prikazuje sliku prostora, kartu prostora robota i poziciju robota. Primjer grafičkog prikaza mreže zauzeća prostora i pozicije robota prikazane su alatom RViz-om na slici 7. Korištenjem različitih senzora i slika prostora, primijećeno je da vremensko izvršavanje jednog ciklusa senzora značajno varira.

6.1. Domet i vidno polje senzora

U radu je moguće mijenjati parametre senzora. Prvi parametar je domet senzora i zadaje se u yaml datoteci. Povećanjem dometa povećavamo broj piksela koje treba provjeriti, i s time se povećava vrijeme jednog ciklusa senzora. U zadanim mjerenjima koristimo metodu emitiranja zraka *raycasting* tako da će se senzori drugačije ponašati uz prisutnost zida i bez prisutnosti. Mjerenja su izvršena s kutom vidnog polja od 90 stupnjeva.

6.1.1. Emitiranje zraka u prostoru bez prepreka

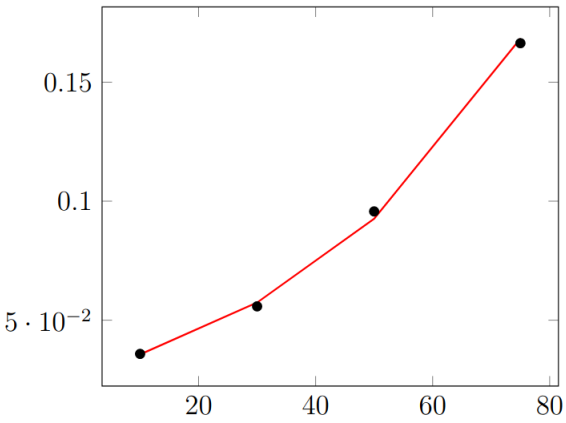
Za zadano mjerenje robot je smješten da mu senzor ne doseže zid. Napravljena su mjerenja za različite radijuse dometa senzora (10, 30, 50, 75). Radijus dometa senzora u tablici je prikazan u pikselima. Tri puta je ponovljeno mjerenje za svaki od radijusa trajanja 20 sekundi i uzeta je srednja vrijednost svih ciklusa.

	10pxl	30pxl	50pxl	75pxl
1.	0.03583	0.05625	0.09799	0.16918
2.	0.03665	0.05562	0.09440	0.16505
3.	0.03635	0.05553	0.09471	0.16494
avg	0.03628	0.05580	0.09570	0.16639

Slika 8: Mjerenja bez zida

Iz mjerenja (Slika 8) se može primjetiti da se povećanjem dometa senzora povećava i vrijeme jednog ciklusa senzora. Povećanje vremena nije linearno već je parabolično. To je

zato što se povećanjem dometa senzora povećava i otkriveni prostor, a taj prostor se povećava kvadratno s obzirom na povećanje dometa senzora. Aproksimirana kvadratna funkcija dobivena iz mjerenja je prikazana na slici 9.



Slika 9: Srednje vrijednosti vremena ciklusa senzora s obzirom na domet senzora

6.1.2. Emitiranje zraka u prostoru s preprekama uz kretanje robota

Eksperimentiranjem je primjećeno da se vrijeme jednog ciklusa senzora samnji ako je senzor detektirao zauzeti prostor. Zato je za drugo mjerenje napravljeno mjerenje ciklusa senzora dok se robot kreće pored zida. Pretpostavka je kada će senzor imati prepreku uz sebe da će mu trebati manje vremena da napravi jedan ciklus zbog smanjenja površine istraživanja.

Kako bi svako mjerenje bilo ispravno, snimljene su komande kretanja u formatu *rosbag* [7]. Robot će svaki put napraviti istu putanju, ali će imati različit domet senzora. Kut senzora u mjerenju je 90 stupnjeva i provedena su tri mjerenja za svaki od radijusa kao i u prethodnom primjeru.

	10pxl	30pxl	50pxl	75pxl
1.	0.03755	0.05018	0.07732	0.13733
2.	0.03791	0.05444	0.07193	0.12338
3.	0.03849	0.05591	0.06875	0.14116
avg	0.03798	0.05351	0.07267	0.13396

Slika 10: Mjerenja sa zidom i kretanjem

Na tablici 10 prikazane su vrijednosti mjerenja. Srednje vrijednosti s manjim radijusom (10 i 30) se nisu značajno promijenile čak se vrijednost s najmanjim radijusom povećala. Pretpostavlja se da je to radi povećanja zahtjeva izvođenja simulacije, a tako mali radijus ne prinosi značajnom ubrzanju ciklusa senzora. Kod većih vrijednosti primjećuju se značajna ubrzanja zbog manje površine potrebne za otkriti.

6.1.3. Vidno polje senzora

Slično kao s dometom senzora, moguće je mijenjati i kut vidnog polja senzora. Napravljeno je mjerenje koje izračunava srednju vrijednost trajanja jednog ciklusa senzora. Odabrani kutevi vidnog polja su 30, 45, 60 stupnjeva (za 90 stupnjeva napravljena su mjerenja na slici 8). Može se pretpostaviti da će povećanjem kuta vrijeme trajanja ciklusa povećati.

	30°	60°	90°	180°
1.	0.05215	0.07382	0.09799	0.14864
2.	0.05152	0.07188	0.09440	0.14895
3.	0.05277	0.07204	0.09471	0.14760
avg	0.05215	0.07258	0.09570	0.14840

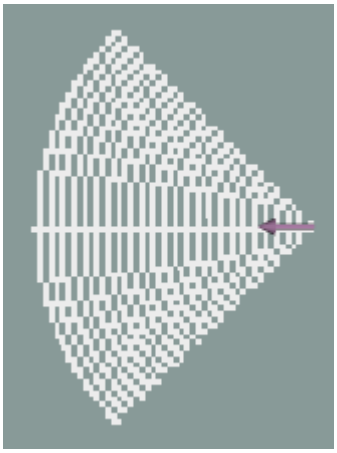
Slika 11: Mjerenja s različitim vrijednosti kuteva

Na slici 11 su prikazane vrijednosti mjerenja i njihova srednja vrijednost za različite kuteve vidnog polja. Zaključujemo da će se povećanjem vidnog polja vrijeme ciklusa senzora povećati.

6.2. Kut zakreta

Korištenjem *raycasting-a*, uz kut vidnog polja i domet senzora, moguće je mijenjati korak zrake i kut zakreta senzora. Korak zrake je zadan na vrijednost jednog piksela i neće se mijenjati u radu. Smanjenjem koraka zrake došlo bi do redundancije jer bi više puta ista zraka ispitivala istu ćeliju.

Npr, ako je veličina koraka senzora pet puta manja od veličine ćelije, moguće je da će ista zraka pet puta ispitati istu ćeliju. Povećanjem koraka zrake bi došlo do neotkrivanja pojedinih ćelija. Na slici 12 prikazan je senzor prostora uz korak veličine dvije ćelije.

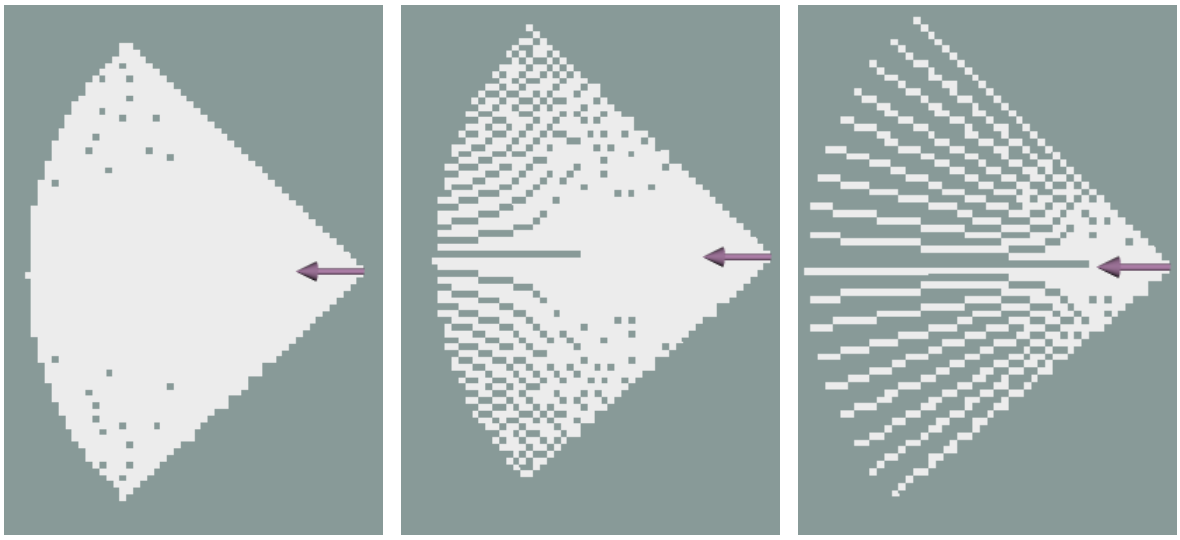


Slika 12: Skenirani prostor korakom zrake veličine dvije ćelije

Veličina koraka senzora postavljena je na veličinu jedne ćelije. Parametar koji će se namještavati je kut zakreta senzora. U početku se za kut zakreta senzora zadavala vrijednost od jednog stupanja. Navedena vrijednost dobra je za male radijuse (do 30 piksela). Ali, povećanjem dometa senzora, zbog prevelikog kuta zakreta gube se ćelije za otkrivanje. Zbog navedenog, kut zakreta će biti obrnuto proporcionalan sa zadanim dometom senzora. Formula kuta zakreta s obzirom na radijus senzora:

$$\phi = k/radijus, \tag{6.1}$$

gdje je ϕ kut zakreta senzora, a k konstanta koja se proizvoljno određuje. U ovom slučaju vrijednost konstante k je određena eksperimentalno i iznosi 25. Povećanjem zadane konstante raste kut zakreta ϕ i gubi se potrebna informacija o skeniranom prostoru. Na slici 13 prikazana je promjena skeniranog prostora ovisna o konstanti koja određuje kut zakreta.



Slika 13: Otkriveni prostor za različite kuteve zakreta ϕ za vrijednosti konstante $k = 50, k = 100, k = 250$

6.3. Karte prostora

Simulacija je isprobana nad različitima kartama slike. Zbog sličnih dimenzija slika prostora nije primjećena promjena vremenskog ciklusa senzora obzirom na sliku karte. Zaključak je da vremenski ciklus senzora ovisi o parametrima senzora, a ne o veličine slike prostora.

Poželjno je da karte prostora nemaju nepoznatog prostora jer nepoznati prostor nema utjecaja na simulaciju. Za simulaciju je najbitniji zauzeti prostor jer tad senzor zna da treba promijeniti zraku, a u nepoznatom prostoru se ponaša kao i u slobodnom. Idealna karta prostora bi bila samo sa slobodnim i zauzetim prostorom.

7. Zaključak

Cilj ovog rada bio je napraviti egzaktnu simulaciju prostora bez korištenja stage simulatora i istovremene lokalizacije i izgradnje karte prostora (SLAM). Naziv egzaktna je dobila što se poznaju svi detalji robotskog sustava i nije potrebno raditi dodatne estimacije poza i prostora. Kako nije potrebno raditi različite algoritme za estimaciju poza i otkrivenog prostora, egzaktna simulacija je napravljena da služi kao bazna simulacija za različite algoritme autonomnog i višerobotnog istraživanja.

Za egzaktnu simulaciju i komuniciranje između korišten je ROS paket. Pomoću ROS-a izrađena je jednostavna komunikacija između robota i slike prostora. Egzaktna simulacija za ulazne podatke dobiva mrežu zauzeća prostora, parametre senzora i poziciju robota. robotski sustav stvara svoju vlastitu kartu prostora....

LITERATURA

[1] A. Batinović, *Decentralizirana strategija kooperativnog istraživanja prostora pomoću višerobotskog sustava*. Diplomski rad, Sveučilište u Zagrebu, 2018.

[2] "ROS." http://wiki.ros.org/ROS, 5. srpnja 2022.

[3] "Opencv, version 4.2.0." https://docs.opencv.org/4.2.0/, 5. srpnja 2022.

[4] "Ros parametri i yaml datoteka." http://wiki.ros.org/rosparam, 5. srpnja 2022.

[5] L. Vandevenne, "Raycasting," 2022. https://lodev.org/cgtutor/raycasting.html.

[6] "RViz." http://wiki.ros.org/rviz, 5. srpnja 2022.

[7] "rosbag." http://wiki.ros.org/rosbag, 5. srpnja 2022.

Egzaktna simulacija autonomnog istraživanja prostora pomoću višerobotskog sustava

Sažetak

Sažetak na hrvatskom jeziku.

Ključne riječi: Ključne riječi, odvojene zarezima.

Title

Abstract

Abstract.

Keywords: Keywords.