Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

Doktorska disertacija

SISTEMI IN KRMILJENJE AVTONOMNIH MOBILNIH ROBOTOV V NEUREJENIH PROSTORIH

Maribor, Februar 2018 Peter Lepej

Avtor: Peter Lepej

Naslov: Sistemi in krmiljenje avtonomnih mobilnih robotov v neurejenih prostorih

UDK: 007.52:004.358(043.2)

Ključne besede: mobilni robot, ujemanje krožnic, sledenje poti, hitrostna regulacija, reševalni robot, brezpilotno leteče vozilo, kombinirana simulacija

Število izvodov: 5

Oblikovanje: Peter Lepej

Lektorica: Nina Koper, prof. slov., uni. dipl. prev. in tolm. za angleški jezik

ZAHVALA

Zahvaljujem se vsem, ki so mi pri študiju in nastajanju doktorske disertacije pomagali in me motivirali. Predvsem se za strokovno pomoč in usmerjanje zahvaljujem svoji mentorici, doc. dr. Suzani Uran.

Zahvala za strokovne nasvete tudi profesorju dr. Geraldu Steinbauerju, vodji projekta TEDUSAR, in raziskovalnemu mentorju v Barceloni, dr. Joanu Solàju. Kolegu in mentorju na področju raziskav avtonomije kmetijskih mobilnih robotov, dr. Juriju Rakunu.

Hvala tudi moji družini, ki mi je stala ob strani ter nudila pomoč in spodbudo med delom na doktorskem študiju.

Univerza v Mariboru

Slomškov trg 15
2000 Maribor, Slovenija

Maribor, 27. 9. 2016
Številka: 414/2016/50/416-MGM

Na osnovi 287., 140., 142. in 144. člena Statuta Univerze v Mariboru (Statut UM-UPB11, Ur. l. RS, št. 44/2015, 92/2015) ter sklepa 14. redne seje Senata Univerze v Mariboru, ki je potekala 27. 9. 2016 v zvezi z vlogo doktorskega kandidata Petra Lepeja za sprejem odločitve o predlagani temi doktorske disertacije in mentorja

izdajam naslednji

SKLEP

Odobri se tema doktorske disertacije Petra Lepeja s Fakultete za elektrotehniko, računalništvo in informatiko z naslovom »Sistemi in krmiljenje avtonomnih mobilnih robotov v neurejenih prostorih«. Za mentorico se imenuje doc. dr. Suzana Uran. Kandidat mora članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih najpozneje do 26. 9. 2020.

Obrazložitev:

Kandidat Peter Lepej je 30. 3. 2016 na Fakulteti za elektrotehniko, računalništvo in informatiko vložil vlogo za potrditev teme doktorske disertacije z naslovom »Sistemi in krmiljenje avtonomnih mobilnih robotov v neurejenih prostorih«. Za mentorico je bila predlagana doc. dr. Suznana Uran.

Senat Fakultete za elektrotehniko, računalništvo in informatiko je na osnovi pozitivnega mnenja komisije za oceno teme doktorske disertacije, ki je ugotovila, da kandidat izpolnjuje pogoje za pridobitev doktorata znanosti, in ocenila, da je predlagana tema ustrezna, sprejel pozitivno mnenje in poslal predlog teme doktorske disertacije s predlogom mentorice v odobritev Senatu univerze.

Senat Univerze v Mariboru je po proučitvi vloge in na osnovi določil Statuta Univerze v Mariboru sprejel svojo odločitev o predlagani temi doktorske disertacije in imenoval mentorico, kot izhaja iz izreka.

V skladu s 144. členom Statuta Univerze v Mariboru mora kandidat za pridobitev doktorata znanosti najpozneje v štirih letih od dneva izdaje tega sklepa članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih. Kandidatu je bil določen rok za oddajo izdelane doktorske disertacije glede na datum sprejetja teme na pristojnem organu.

Pouk o pravnem sredstvu:
Zoper ta sklep je možna pritožba na Senat Univerze v Mariboru v roku 8 dni od prejema tega sklepa.

Rektor Univerze v Mariboru:
Prof. dr. Igor Tičar

UNIVERZA V MARIBORU

Obvestiti:
1. Kandidata.
2. Fakulteto.
3. Arhiv.

KAZALO

SEZNAM SLIK

SEZNAM TABEL

SEZNAM KRATIC IN POJMOV

BLV	brezpilotno leteče vozilo (angl. *unmanned aerial vehicle*)
DPKL	dinamično prilagajanje krožnih lokov (angl. *dynamic arc fitting*)
DWA	pristop dinamičnega okna za namen načrtovanja in izvajanje poti (angl. *dynamic window aproach*)
GPS	globalni sistem pozicioniranja
HIL	simulacija, ki vključuje realne elemente (angl. *hardware-in-the-loop*)
IME	inercijska merilna enota
LED	svetleča dioda (angl. *light-emitting diode*)
LMR	laserski merilnik razdalje
NIST	Nacionalni inštitut za standardizacijo in tehnologijo (angl. *National Institut of Standards and Technology*)
OS	opazovalec sistema, v smislu spremljanja toka programskega toka podatkov
P	proporcionalni regulator
PI	proporcionalno – integrirni regulator
PID	proporcionalno – integrirni – diferencirni regulator
preiskovanje	prevod angleške besede *exploration*, kar v mobilni robotiki predstavlja preiskovanje, raziskovanje prostora

RGB	barvni model R (red – rdeča), G (green – zelena), B (blue – modra), vsaka od teh barv se lahko pojavi v 256 odtenkih, kar skupno znaša 16.777.216 barv
RGBD	barvni model RGB z dodatnim parametrom D, ki predstavlja razdaljo (angl. *distance*)
ROS	robotski operacijski sistem (angl. *robot operating system*)
SIŽ	sistem za iskanje žrtev
SLAM	simultana lokalizacija in kartiranje (angl. *simultaneous localization and mapping*)
TCP/IP	mrežni komunikacijski protokol**,** ki podpira TCP, ima transportni osebek TCP (v jedru operacijskega sistema ali kot uporabniški proces), ki upravlja tokove TCP in vmesnik do sloja IP.
TEDUSAR	Tehnologije in izobraževanje za reševanje in iskanje z roboti (angl. *Technology and Education for Search and Rescue Robot*)
Kvadrokopter	leteče mobilne naprave s štirimi veternicami
X-Y ravnina	prostostna ravnina, ki je definirana vodoravno
YUV	barvni model je mogoče izračunati iz RGB barvnega formata, ta oblika je bolj prijazna človeku in predstavlja (Y - osvetljenost) in (UV – barvna komponenta)
2D	dvodimenzionalen prostor z dvema osema X in Y
3D	tridimenzionalen prostor s tremi osmi X, Y in Z

SEZNAM OZNAK

$\boldsymbol{b}$	razdalja med pogonskima kolesoma mobilnega robota, pravokotno na smer potovanja
$\boldsymbol{SV}$	središče vrtenja ali os vrtenja
$\boldsymbol{Fk}, \boldsymbol{Fg}$	sile, ki delujejo med tlemi in kolesom ali gosenico
$\boldsymbol{Pvn}$	pogled vnaprej
$\boldsymbol{R}$	polmer krožnice
$\boldsymbol{Rg}$	premer gosenice
$\boldsymbol{Rk}$	premer kolesa
$\boldsymbol{v}$	splošno linearna ali translacijska hitrost
$\boldsymbol{vl}, \boldsymbol{vr}$	hitrosti levega in desnega kolesa, definirane glede na ravnino X-Y
$\boldsymbol{vk}, \boldsymbol{vg}$	linearna hitrost kolesa in gosenice
$\boldsymbol{w}$	splošno kotna hitrost
$\boldsymbol{wk}, \boldsymbol{wg}$	kotna hitrost kolesa in gosenice
$\boldsymbol{Xs}, \boldsymbol{Ys}, \boldsymbol{\theta}$	položaj in orientacija mobilnega robota v prostoru, definiran v globalnem koordinatnem sistemu
$\boldsymbol{\omega_l}, \boldsymbol{\omega_r}$	kotne hitrosti levega in desnega kolesa

SISTEMI IN KRMILJENJE AVTONOMNIH MOBILNIH ROBOTOV V NEUREJENIH PROSTORIH

UDK: 007.52:004.358(043.2)

Ključne besed: mobilni robot, ujemanje krožnic, sledenje poti, hitrostna regulacija, reševalni robot, brezpilotno leteče vozilo, kombinirana simulacija

Povzetek: Avtonomna mobilna robotika je uporabna v nadzornih sistemih in kot pomoč pri iskanju in reševanju žrtev v primeru nesreč. Regulacija hitrosti modularnega avtonomnega sistema za pomoč pri reševanju ob naravnih nesrečah ima veliko težo, saj je bistvena za pravilno gibanje v prostoru. Ena izmed pomembnih nalog v avtonomni robotiki je regulacija hitrosti (linearne in kotne), ki jo proizvede avtonomni sistem za mobilno bazo. Naš glavni cilj je ustvariti hitrostni regulator na višjem nivoju, ki ga je mogoče uporabiti na različnih mobilnih platformah. V tem primeru se osredotočamo na mobilne robote z diferencialnim pogonom. Naš pristop sloni na iskanju ujemanja krožnic, ki upošteva kinematični model in omejitve mobilne baze robota. Na podlagi algoritma se izvede trajektorija, ta je odvisna od nastavljenih parametrov, ki so prilagojeni za mobilno bazo. Dodatno smo v algoritem vključili poravnavo kota orientacije in kompenzacijo naklona kota med gibanjem mobilnega robota. Opisali smo našega reševalnega robota in sisteme za avtonomno delovanje in implementacijo algoritmov. Predlagan algoritem za sledenje smo primerjali z algoritmom Hector. Prikazani so rezultati primerjav in rezultati sledenja mobilnega robota v simulacijah in realnih eksperimentih. Nato smo nadaljevali naše delo z implementacijo našega algoritma na brezpilotnih letečih vozilih, kjer smo zgradili kompletno simulacijo z možnostjo vključitve pravega letečega vozila. Na tem sistemu smo izvedli serijo eksperimentov, da bi dokazali koncept sistema.

SYSTEMS AND CONTROL FOR AUTONOMOUS MOBILE ROBOTS IN CLUTTERED ENVIRONMENTS

UDK: 007.52:004.358(043.2)

Keywords: mobile robot, arc fitting, path following, velocity control, rescue robot, UAV, hardware in the loop

Abstract: Many applications such as surveillance, inspection, search and rescue operations can be performed with autonomous robots. Our aim is to control a modular autonomous system in rescuing robotics. One of the basic problems regarding autonomous robotics is the execution part in which the control commands (translation and rotational velocities) are produced for mobile bases. We have focused on this area because for skid-steered mobile robots there is available only a small amount of path following software. Our goal was to develop a velocity controller that could be used for multiple skid-steered mobile bases. We considered differential drive mobile bases such as tracked skid-steering mobile base. Our approach is based on an arc fitting algorithm which takes into account the robot constrains and kinematical model. It produces continuous trajectory where fitting to the given path depends on given parameters adapted to the mobile base. Moreover, we have included orientation angle compensation while the mobile robot is moving, and ground inclination compensation. Our rescue robot is described together with the simulation setup and algorithm implementation. We compared our algorithm to the Hector-based software. We showed the results of the compared algorithms and the results of mobile robot path following in simulation and real experiments. Later on we used our algorithm on an unmanned aerial vehicle platform; we have built a simulation in which real aerial vehicle could be included. On this system setup we performed a series of experiments to prove the concept.

1 UVOD

Na splošno avtonomni mobilni sistem predstavlja zaključeno celoto, ki je zmožna samostojnega odločanja. Samostojno odločanje se izvaja pod določenimi pogoji in z danim ciljem. Za dosego danega cilja avtonomni sistem uporablja vse razpoložljive informacije. Razpoložljive podatke predstavljajo izhodi senzorjev v različnih digitalnih oblikah; v obliki slik (dvodimenzionalni) ali informacije v obliki razdalje (enodimenzionalni). Avtonomni mobilni sistem skuša uporabiti vso znanje, podano iz senzorjev, in s tem zgraditi digitalni model okolja. Sledijo navigacijski algoritmi, ki omogočajo različne operacije v zgrajenem modelu okolja.

V tem delu se bomo posvetili problematiki avtonomnega mobilnega sistema v obliki kopenskega in letečega mobilnega robota. Raziskovalno delo zajema študijo in pregled obstoječe literature, dostopne v lokalnih in mednarodnih spletnih bazah, ter splošno dostopnega gradiva. Predstavljena doktorska disertacija vsebuje tudi študije katere smo objavili v obliki člankov. Vključuje analitične izpeljave ter dokaze in preverjanja na podlagi simulacijskih, realnih eksperimentov in kombinacijo obeh. Preučili in implementirali smo nekaj potencialno najprimernejših rešitev, ki jih bomo v nadaljevanju analizirali ter izpostavili njihove prednosti in slabosti. Prednosti bomo vključili v našo rešitev algoritma za sledenje poti.

Metode raziskovalnega dela, uporabljene v doktorski disertaciji, zajemajo:

Simulacije: za preizkušanje zgrajenih sistemov in algoritmov (neodvisno od mehanskih in napajalnih omejitev) bomo zgradili simulacijsko okolje za reševalne robote, kjer lahko neomejeno preizkušamo razvite algoritme. Za namen testiranja in raziskovanja sledenja poti z brezpilotnim letečim vozilom (angl. *unmanned aerial vehicle* ali krajše BLV) v neurejenem okolju bomo prav tako zgradili simulacijsko okolje.

Kombinirana simulacija (angl. *hardware-in-the-loop* ali krajše HIL): v primeru lokacijskih in mehanskih omejitev smo uporabili sistem, ki je vpeljal pravega robota v simulacijsko okolje. To je za mobilnega robota v primeru izvajanja eksperimentov virtualna resničnost; pričakujemo lahko bolj zanesljive in realistične rezultate eksperimentov.

Realni eksperimenti: realne eksperimente z reševalnim mobilnim robotom smo izvajali v simulirani reševalni areni NIST v laboratoriju za kognitivne sisteme v mehatroniki, in sicer v areni NIST laboratorija v Gradcu, ter v arenah NIST med sodelovanjem na RoboCup tekmovanjih v robotskem reševanju (RoboCup German Open 2012 in 2013, ter svetovnem tekmovanju RoboCup 2014 v Braziliji). Naš robot je bil med

prvimi goseničnimi roboti, ki je avtonomno raziskoval reševalno areno. Prav tako smo robota večkrat preizkusili v urbanem okolju.

1.1 OPREDELITEV PROBLEMA IN MOTIVACIJA

Mobilna robotika v vsej svoji razsežnosti zajema avtonomne naprave oziroma stroje, ki so zmožni opravljati določene mobilne operacije samostojno. V tem delu bomo obravnavali kopenske avtonomne mobilne robote (na kolesih ali gosenicah), ki imajo diferencialni pogon, ter leteče avtonomne mobilne robote. Namen takšnih robotov je zelo širok, zato se bomo v delu osredotočili le na avtonomno delovanje mobilnih robotov, namenjenih uporabi na področju reševanja v neurejenem in zahtevnem okolju. Kot primer zahtevnega, neurejenega in nepredvidljivega okolja lahko navedemo naravo, kot primer nevarnega okolja pa lahko izpostavimo ruševine. Gibanje kopenskih mobilnih robotov lahko obravnavamo v 2D- ali 2,5D-okolju, ko se usmerimo na leteče robote, pa moramo upoštevati okolje v treh dimenzijah. Motivacija za obravnavanje tematike avtonomnega reševanja z mobilnimi roboti je predstavljena v nadaljevanju.

Potresi spadajo med naravne nesreče, ki jih ni mogoče točno napovedati, vseeno pa jih z instrumenti lahko zaznavamo, merimo njihovo moč ter zbiramo podatke o njihovi moči in pogostosti. V Sloveniji skrbi za beleženje in obveščanje o potresih Agencija Republike Slovenije za okolje, ki na svojih spletnih straneh objavlja podatke o potresih, nastalih v Sloveniji, in izdeluje karte potresne ogroženosti Slovenije na osnovi zbranih podatkov. Po podatkih svetovne statistike se v enem letu na Zemlji v povprečju zgodi 1 (ali vse številke piši s številko ali vse z besedo) potres z magnitudo 8 ali več, 15 potresov z magnitudo 7 do 7,9, 134 potresov z magnitudo 6 do 6,9 in 1319 potresov z magnitudo 5 do 5,9. V letu 2012 je našo pozornost vzbudilo kar nekaj potresov; eden takšnih je bil potres z magnitudo 6,3, ki se je zgodil marca istega leta v bližini severovzhodne obale otoka Honšu na Japonskem in je poškodoval jedrsko elektrarno v Fukušimi. Drugi javno odmevni potresi, pa so nastali v naši neposredni bližini. Od 20. maja 2012 do 3. junija 2012 se je na območju severne Italije namreč zgodilo več potresov z magnitudo od 5 do 6; nekatere izmed potresnih sunkov smo občutili tudi v Sloveniji. V Italiji so se zaradi posledic potresa rušile stavbe, potres je terjal več človeških življenj, mnogi ljudje so morali zapustiti svoje domove ...

Zgoraj naštete posledice teh naravnih katastrof so motivirale Japonce, da so pričeli z organizacijo tekmovanja RoboCup. To tekmovanje je precej razsežno, zato se bomo osredotočili na reševalno (angl. *rescue*) sekcijo RoboCup tekmovanja. Cilj robotskega tekmovanja RoboCup, ki vključuje reševalne robote, je zgraditi robote, ki bi pomagali reševalcem pri iskanju morebitnih žrtev v primeru potresa. Uspešno in učinkovito reševanje, ki zagotavlja rešitev čim večjega števila ponesrečencev, mora namreč potekati hitro.

To lahko dosežemo z dovolj velikim številom reševalcev, ki pa jih v večini primerov nimamo dovolj, ta manko pa bi se lahko nadomestil z reševalnimi roboti. V primeru reševanja ljudi po potresu bi morali reševalni roboti avtonomno raziskovati neznane prostore in hkrati iskati znake življenja. Za raziskovanje neznanih prostorov bi morali imeti roboti laserske merilnike razdalj (LMR) in algoritme, ki izdelujejo 2D- in 3D-zemljevide. Znake življenja v danem prostoru bi roboti za reševanje iskali s pomočjo RGB-kamer, termo kamer in drugih podobnih senzorjev. Trenutno se RoboCup tekmovanja iz reševanja odvijajo v standardnih arenah NIST, kjer je simulirano zahtevno popotresno in neurejeno okolje. Za sodelovanje na tekmovanju so potrebni avtonomni roboti, ki zmorejo sami neodvisno operirati v neurejenih okoljih. Podrobnejši opis robotov za reševanje po potresu je mogoče najti v [1].

Za delo z letečimi ali kopenskimi roboti smo se odločili z namenom, da bi pomagali pri reševanju v primeru naravnih nesreč. Mobilni leteči robot je v primerjavi s kopenskim hitrejši in mu ovire na tleh, stopnice, nadstropja …, ne predstavljajo težav. Leteče robote z navigacijskim sistemom, učinkovitim pri reševanju, pa lahko uporabimo tudi za izvajanje nadzornih nalog, kot so pregled kmetijskih zemljišč, ocenjevanje stanja gozdov, preglede na težko dostopnih področjih in podobno.

Pri gradnji avtonomnih reševalnih robotov je treba obravnavati problematiko, ki jo tukaj samo na kratko povzemamo z namenom predstavitve pojmov, ki jih bomo uporabljali v nadaljevanju, ko sledijo tudi podrobnejši opisi.

1. **Kartiranje in lokalizacija**

   Izdelovanje 2D-/3D-zemljevida trenutne okolice s pomočjo senzorjev, kot so LMR, RGBD kamere in podobni. Ti omogočajo hkratno določanje položaja v izdelanem zemljevidu (s pristopom SLAM – angl. *simultaneous localization and mapping*).

2. **Preiskovanje:**

   Glede na dano okolico, ki je zajeta s pomočjo senzorjev, se mora robot avtonomno odločiti, v katero smer bo raziskoval okolico. Ta strategija se izdeluje na podlagi zemljevida (SLAM). Posebna tehnika preiskovanja je obravnavana v [2], kjer mobilni robot uporabi vse senzorske podatke iz okolja, na podlagi česar se izvede selektivno preiskovanje.

3. **Načrtovanje poti**

Poteka na podlagi izbranega zemljevida in senzorjev, ki lahko zaznajo dinamične ovire na robotovi poti. Načrtovanje se izvede na podlagi velikosti modela robota, okolice, zmožnosti robota in izdelanega zemljevida (SLAM).

4. **Izvedba ali sledenje poti**

Pomembno je, da se pot gibanja robota tudi izvede tako, kot je bila načrtovana. Za to skrbi nižji nivo, ki je prilagojen dani mobilni bazi. Na nižjem nivoju vodenja mobilnega robota krmilimo s translacijsko in kotno hitrostjo. Ti se nato preračunata v hitrost vrtenja gosenic ali koles in se izvedeta s pomočjo hitrostne regulacije. Slednja je na najnižjem nivoju v večini primerov izvedena s pomočjo dajalnikov pozicije in P, PI ali PID regulatorja. Kljub regulaciji hitrosti gosenic premikanje robota po načrtovani poti ni zagotovljeno, saj lahko prihaja med vožnjo do zdrsov gosenic, ki jih je mogoče kompenzirati le s pomočjo LMR in zemljevida. Ker nismo našli primernega in učinkovitega algoritma za izvajanje dane poti za našega goseničnega robota, smo se odločili za podrobnejšo obravnavo tega problema.

5. **Iskanje žrtev**

Za učinkovito zaznavo žrtev imamo na voljo različne senzorje, ki so nameščeni na senzorski glavi robota. Prvi korak pri iskanju žrtev je iskalna strategija, ki mora biti prilagojena robotski bazi, in število prostostnih stopenj senzorske glave. Naslednji korak je procesiranje danih signalov in prepoznava znakov življenja. V tem primeru smo se lotili mehanske konstrukcije senzorske glave in iskalne strategije, ter naprednejših algoritmov za prepoznavo potencialnih žrtev.

6. **Samostojnost robota**

V številnih primerih se robot ustavi zaradi kakšne težave na programski ali strojni opremi. Problemi, kot na primer napačno delovanje senzorjev ali zatik servomotorjev, se lahko pojavijo tudi med izvajanjem naloge. Ker se reševalni roboti gibljejo po nevarnem prostoru, je pomembno, da imajo zmožnost diagnosticiranja in samostojne odprave manjših napak med izvedbo naloge reševanja.

Pri našem delu smo se srečali z vsemi omenjenimi problemi, vendar za večino izmed njih obstajajo različne rešitve in so dobro poznane v robotski skupnosti. Naslovili smo sledeče probleme: izvajanje poti v 2D in 3D prostoru, problem natančne izdelave zemljevida in samostojnost avtonomnega mobilnega robota. Pri iskanju rešitve za sledenje poti nismo našli primernega algoritma sledenja za goseničnо izvedbo robotov,

zato smo predlagali svojega. Izvedba poti je odvisna od kvalitete zemljevida, ki ga gradi robot, zato je bilo potrebno izboljšati sistem za kompenzacijo razgibanosti tal. Pri delu z roboti, ki imajo več vgrajenih sistemov, so napake v sistemih zelo pogoste, kar nas je motiviralo tudi za iskanje rešitve tega problema. Delo na mobilnih robotih smo začeli v okviru projekta TEDUSAR (**Tehnologije in izobraževanje za reševanje in iskanje z roboti)** na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru in Tehniški univerzi v Gradcu ter ga nadaljevali na Inštitutu za robotiko in industrijo v Barceloni. V Španiji smo se posvetili razvoju navigacijskega sistema za letečega robota (BLV), njegova naloga je bila navigacija v neurejenem okolju s pomočjo 3D LMR.

Osnovni problem v mobilni robotiki je nasproten problemu človeka (kjer se omejimo na manjše prostore), saj človek načeloma nima težav s prepoznavo okolice in lokalizacije v njej. Človek po t.i. samodejni lokalizaciji v prostoru načrtuje gibanje, hojo in pot do želenega cilja. Robot ima v nasprotju človekom osnovno težavo v dojemanju okolja, saj so senzorji in algoritmi za lokalizacijo pretežno togi, počasni in imajo povečano možnost napake. Nasprotno, pa je načrtovanje v zaznanem okolju na današnjih procesorskih sistemih relativno hitro.

1.2 CILJI DOKTORSKE DISERTACIJE

Za potrebe reševalne naloge, opisane v poglavju 1.1, kjer je potrebno gibanje v neurejenem in nepredvidljivem okolju, je bilo potrebno zgraditi robota. Zgradili smo dva reševalnega robota na osnovi dveh različnih robotskih baz - Dr Robot Jaguar V1 (krajše Jaguar) in Mesa Element (krajše Mesa). Za polno funkcionalnost omenjenih robotov je bilo potrebno razviti tudi sisteme za navigacijo in sistem za iskanje žrtev.

Najbolj priljubljeni in najnovejši senzorji za zajem okolja so LMR, merijo namreč razdaljo do ovir v prostoru v 1D in 2D ravninah. LMR je primeren za uporabo v mobilni robotiki, saj deluje zelo hitro (10Hz) in natančno (milimetrska natančnost). Na podlagi teh meritev lahko nato izdelamo zemljevid okolja s pomočjo postopkov za hkratno lokalizacijo in gradnjo zemljevida okolja (angl. *simultaneous localization and mapping* ali na kratko SLAM). Na podlagi izdelanega zemljevida se izračunavajo nove raziskovalne poti v prostoru. Nato pa mora robot načrtovano pot v prostoru tudi uspešno izvesti. Izvedbo sledenja načrtovane poti smo poskusili zagotoviti s pomočjo obstoječega algoritma dinamičnih oken (angl. *dynamic window approach*, krajše DWA), ki je priljubljen v mobilni robotiki. Prav tako smo preizkusili tudi algoritem za sledenje poti kateri je vključen v navigacijski programski paket Hector (angl. *hector_path_follower*), ki ga uporablja ena izmed najboljših tekmovalnih ekip (TU Darmstadt 2013 in 2014) na tekmovanju »RoboCup

reševanje«. V primeru goseničnega mobilnega robota sta bila oba algoritma pri avtonomnem sledenju poti v neurejenem razgibanem okolju neuspešna. Sledenje poti robotskih baz z diferencialnim pogonom in gosenicami je pogojeno z regulacijo hitrosti gosenic. Nižje nivojska regulacija hitrosti gosenic daje pričakovano boljše rezultate od baze brez regulacije hitrosti gosenic. Zaradi zdrsov gosenic med vožnjo robota pa je za učinkovito sledenje poti potreben še dodaten algoritem za kompenzacijo odstopanja od načrtovane poti. Zato smo si zastavili cilj, da za obe robotski bazi razvijemo algoritem, ki bo enostaven, hiter in učinkovit. Na osnovi razvitega algoritma smo zapisali hipotezo **H1**.

Hipoteza 1 (H1): Predstavljen algoritem za sledenje poti z mobilnim robotom temelji na osnovi dinamičnega prilagajanja krožnih lokov gibanja mobilnega robota (krajše DPKL). Predvidevamo, da bo omenjeni algoritem omogočal prilagajanje translacijske in kotne hitrosti glede na trenutno okolje, ter da bo mobilni robot, glede na podane pogoje hitrosti in zmogljivosti mobilne baze, zanesljivo sledil zastavljeni poti v prostoru. **(H1)**

Opazili smo težave pri gradnji zemljevida s SLAM, ki so posledica omejitev pri izravnavanju LMR. Izkušnje ekip lige »RoboCup reševanje« iz preteklih let, npr. Hector idr., so pokazale, da je za uspešno izvedbo SLAM, potreben sistem za izravnavanje 2D LMR. Obstoječi sistemi za uravnavanje so bili omejeni s kotnimi zasuki in so lahko kompenzirali samo določene ovire na poti robota.

Hipoteza 2 (H2): Predpostavljamo, da bo predlagani elektro-mehanski sistem realno časovno kompenziral naklone (po oseh X in Y) robotske baze. Naklone robotske baze povzročijo geometrije površine po kateri se robot premika, kot so ovire, stopnice in luknje. Predvidevamo da bo omenjen sistem uspel kompenzirati naklone večjih klancev, ki dosegajo naklon +/- 45°. Nov mehanski sistem za prilagajanje vodoravnosti LMR naj bi bo zagotovil, da bo izdelan zemljevid natančnejši od sistema brez prilagajanja LMR. **(H2)**

Nevarno okolje pri reševanju zahteva visoko zanesljivost delovanja reševalnih sistemov, zato so nujni tudi postopki za samo diagnozo in odpravljanje manjših napak. Da bi povečali zanesljivost delovanja mobilnega robota, se uporabi diagnostični sistem. Diagnostični sistem kateri spremlja podatkovne tokove in njihovo ponovljivost že obstaja, kar pa v večini primerov ne obsega fizičnega nivoja. Spremljanje indikatorjev pravilnega delovanja bi bil velik doprinos k skupnemu diagnostičnemu sistemu. V osnovi je diagnostični sistem sestavljen iz programskih opazovalcev in upravljalca kateri reagira v primeru zaznanih napak, tukaj predstavljamo novost opazovalca na fizičnem nivoju kar smo poimenovali diagnostična plošča. Diagnostična plošča predstavlja pomemben del diagnostičnega sistema saj bi z opazovanjem podatkov na fizičnem nivoju prej in bolj zanesljivo opazili nepravilnosti v delovanju sistemov.

Hipoteza 3 (H3): Predlagan sistem diagnostične plošče bo spremljal delovanje podsistemov mobilnega robota z opazovanjem odstopanja različnih parametrov, kot so napajalna napetost in tok, količina podatkovnega prometa, odzivnost in podobno. Predpostavimo, da bo sistem diagnostične plošče omogočal priklop večih podsistemov in spremljanje parametrov vsaj 10-tih kanalov hkrati ter da bo za vsak kanal posamezno omogočal vklop/izklop podsistema s strani programskega upravljalca. Omenjen sistem diagnostične plošče bo prav tako omogočal izvajanje samo popravljivih scenarijev v primeru nepravilnega delovanja strojne opreme. Naša predvidevanja so, da bo diagnostična plošča skupaj s programsko opremo omogočala uspešno popravilo napak, ki se lahko pojavijo med delovanjem avtonomnega mobilnega robota. Napake, katere se lahko pojavijo med avtonomnim delovanjem, so nepredvideni zatiki servomotorjev, zagozditev pogonskih motorjev, nepravilno delovanje senzorskih sistemov in podobno. **(H3)**

Naše delo bomo nadaljevali z letečimi roboti, ki prav tako lahko raziskujejo neurejeno okolico, kot so na primer ruševine. Za to zahtevno nalogo je smiselno zgraditi simulator in nato izvesti realni preizkus. Za avtonomno navigacijo je namreč nujno uporabiti sistem za izdelavo zemljevida ter zgraditi sisteme za preiskovanje, načrtovanje in izvedbo poti z letečim robotom. Iz obstoječih navigacijskih principov bomo zgradili avtonomni navigacijski sistem za leteče robote. Sestavni del avtonomnega navigacijskega sistema letečega robota bo algoritem, ki smo ga razvili za sledenje poti z dinamičnim prilagajanjem krožnih lokov (DPKL).

Hipoteza 4 (H4): Predpostavljamo, da bo leteči avtonomni mobilni robot lahko samostojno preiskoval izbrano okolico, s pomočjo algoritma DPKL, sestavljeno iz kompleksnih struktur. Z integracijo našega na novo razvitega algoritma za sledenje, želimo dokazati široko uporabnost slednjega. Algoritem smo vgradili v leteči avtonomni mobilni robot, predvidevali smo, da bo sposoben samostojno raziskovati okolico in zanesljivo slediti podani poti tudi v 3D prostoru. **(H4)**

1.3 IZVIRNI PRISPEVKI

V jedru tega dela predstavljamo algoritem, ki s pomočjo krožnega gibanja dinamično prilagaja sledenje načrtani poti. Algoritem temelji na izračunu krožnice, ki predvidi ukaze za potovanje robota v idealnih pogojih okolice. Ker pa je okolica v neurejenih pogojih spremenljiva, predstavljen algoritem z visoko hitrostjo prilagaja parametre sledenja in s tem ustvari gladko trajektorijo sledenja. Algoritem bo primeren za robote z diferencialnim pogonom s kolesi ali gosenicami. V raziskavi pričakujemo naslednje izvirne znanstvene prispevke:

- Nov algoritem bo primeren za baze mobilnih robotov, ki jih bo mogoče hitro in enostavno prilagoditi in uporabiti za sledenje načrtane (začrtane) poti. Algoritem namreč uporablja krožno gibanje mobilnega robota, ki omogoča osnovno gibanje, od vrtenja na mestu do premočrtnega gibanja mobilne baze. Ta višje nivojska regulacija mobilne baze bo v pomoč pri uporabi različnih avtonomnih mobilnih sistemov.

- V pomoč mobilni robotski bazi za izdelavo natančnega zemljevida smo zasnovali nov in izboljšan sistem za uravnavanje laserskega merilnika razdalje, prav tako smo naredili analizo odzivnosti. Nov sistem za uravnavanje LMR bo omogočal gibanje mobilnega robota tudi po stopnicah, predvsem zaradi večje širine prostostnih stopenj in odzivnosti na trenutni teren.

  Kompleksni sistemi, kot je avtonomni mobilni robot, potrebujejo nekakšen nadzorni sistem. Omislili smo si diagnostični sistem, katerega sestavlja mehansko-električna komponenta in programski del. V kombinaciji delovanja je omogočeno spremljanje parametrov delovanja sistemov mobilnega robota in v primeru napak ali nepravilnosti delovanja tudi ukrepati.

- Kopenski mobilni robot ima omejeno mobilnost, zato smo si v nadaljevanju naše študije želeli uporabiti letečega avtonomnega mobilnega robota. Za omenjen sistem smo razvili navigacijske algoritme in sistem vodenja mobilnega robota v kompleksnih okoljih. Integrirali smo tako imenovano navidezno resničnost, kjer smo resničnemu mobilnemu robotu v digitalnem svetu predstavili okolje, v katerem se je moral znajti. Navidezna resničnost je bila izbrana predvsem zaradi varnosti in drage opreme, za katero neuspeli test v realnih pogojih v večini primerov pomeni uničenje.

1.4 STRUKTURA DISERTACIJE

V uvodnem delu disertacije predstavimo motivacijo za naše delo in opišemo raziskovalne tehnike. V drugem poglavju sledi predstavitev mobilnih robotov, podrobneje mobilni gosenični roboti, katerega smo zgradili skupaj s sistemi za avtonomijo. Opisani so kompleksni elektro-mehanski sistemi mobilnega robota, ki so sestavni deli avtonomije mobilnih robotov. Prav tako so predstavljeni sistemi za uravnavanje laserskega merilnika razdalje, sistem senzorske glave in diagnostična plošča. V tretjem poglavju je obravnavan programski del avtonomnega sistema, ki se osredotoča na področja naprednih algoritmov. Omenjeni algoritmi so hierarhično povezani v celoto. Tukaj se osredotočimo na sledenje poti in

podrobneje raziščemo tehnike sledenj s krožnicami. Četrto poglavje podrobneje opiše naše osrednje delo na področju avtonomnega sledenja z mobilnim robotom. Predstavljeni so rezultati, ki smo jih dobili s pomočjo naših in drugih algoritmov na področju sledenja poti. Prav tako smo v naše delo vključili sisteme, ki so bili razviti v okviru dela na brezpilotnih letečih robotih v petem poglavju, kjer na novo razvit algoritem vključimo in preizkusimo v 3D prostoru.

2 AVTONOMNI MOBILNI ROBOT

Podrobnejša študija obstoječega stanja avtonomnih mobilnih robotov za reševanje je predstavljena v osebni bibliografiji [26], kjer je opisano trenutno stanje tehnologije za reševanje in vključevanje avtonomnih mobilnih robotov v ta segment. Predstavljene so baze avtonomnih mobilnih platform, ki so lahko kopenske, leteče ali podvodne. Vsem mobilnim platformam je skupno to, da so v večini primerov manjših dimenzij – predvsem zaradi ekonomičnosti v fazi razvoja. Podrobnejšo sestavo avtonomnih mobilnih robotov za reševanje smo raziskovali s pomočjo opisov robotov, ki sodelujejo na tekmovanju RoboCup (angl. *team description paper* ali krajše TDP) [87].

Avtonomni reševalni mobilni roboti v večini primerov uporabljajo kolesa ali gosenice za premagovanje ruševin. Nekateri reševalni mobilni roboti za lažje premagovanje stopnic in večjih ovir poleg gosenic ali koles imajo nameščene dodane ročice, katere so lahko opremljene z goseničnim pogonom. Naslednji osnovni sestav avtonomnih mobilnih robotov predstavljajo LMR v različnih izvedbah, za izdelavo zemljevida okolice. V zadnjih časih veliko ekip uporablja tudi sistem za uravnavanje LMR. Sledijo senzorski sistemi, ki jih sestavljajo kamera, termalna kamera, v nekaterih primerih tudi mikrofoni, ter senzorji nevarnih plinov. Senzorski sistemi omogočajo mobilnemu robotu prepoznavanje znakov življenja med potovanjem in s tem identifikacijo potencialnih žrtev med ruševinami. Programska oprema takšnih mobilnih robotov v večini primerih bazira na ROS operacijskem sistemu. Naše reševalne mobilne robote smo gradili na osnovi preverjenih rešitev, ki smo jih želeli nadgraditi in izboljšati.

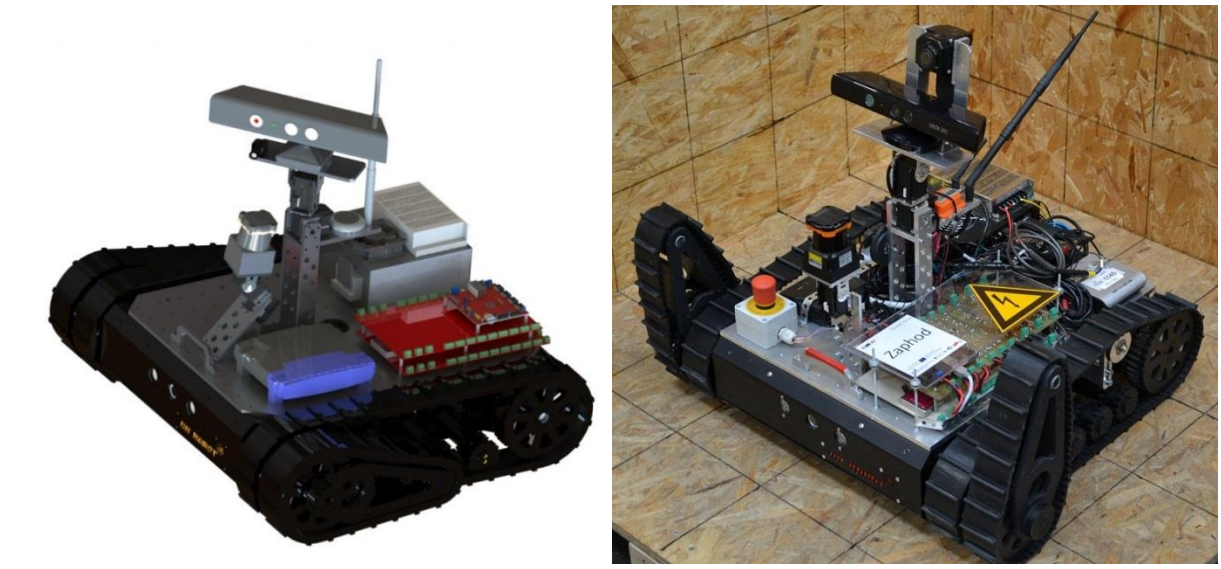
Avtonomne kopenske mobilne robotske baze, uporabljene v našem delu, so opremljene z mehanskimi in senzorskimi sistemi, ki predstavljajo osnovo za avtonomno delovanje. Vsi sistemi so nameščeni na mobilno robotsko bazo. V omenjenem delu predstavimo tudi najnovejše senzorje, in sicer termalno kamero 2D in 3D LIDAR senzorje, ter nekaj primerov uporabe reševalnih robotov. Avtonomnega mobilnega robota sestavljajo štirje glavni sistemi. Prvi je mobilna platforma; to je osnovna platforma za premikanje in pritrditev sistemov za avtonomno delovanje. Sledijo: sistem za uravnavanje laserskega merilnika razdalje,

sistem senzorske glave in diagnostična plošča. V nadaljevanju so opisane mobilne baze s pripadajočo kinematiko ter senzorskimi sistemi.

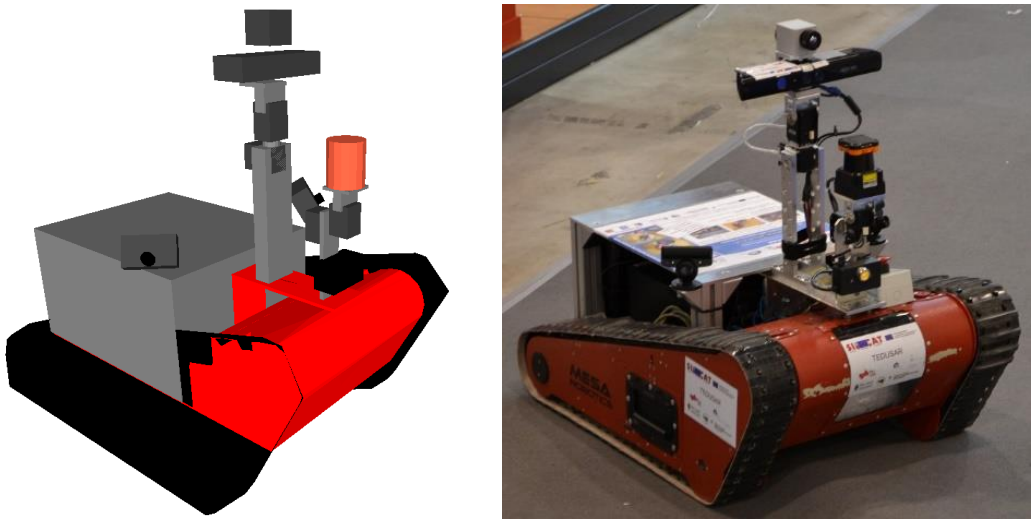
2.1 MOBILNA ROBOTSKA BAZA

Naše delo je slonelo na dveh mobilnih robotskih bazah na gosenični pogon, prva platforma proizvajalca Dr Robot z imenom Jaguar V1, druga delo proizvajalca MESA z nazivom Element. Mobilna platforma Jaguar je zgrajena iz kompaktne plastike, osi so kovinske in ohišje robota je narejeno iz aluminijaste pločevine. Robot ima vgrajen IME (inercijsko merilno enoto), sistem globalnega določanja položaja (GSP) in kompas. Iz senzorja IME lahko pridobimo podatke o naklonu robotske baze med gibanjem. Mobilni robot Mesa pa je zgrajen iz kovinskega ohišja z vgrajenimi NiCd baterijami in koračnimi motorji z zobniškimi prenosi in krmilnikom. Za premikanje uporablja gumijaste gosenice ter ima nosilnost do 55 kg. V osnovi nima vgrajenih senzorjev, razen senzorja zasuka motorjev-koles.

Mobilna robotska baza Jaguar na nižjem nivoju ne uporablja regulacijskih zank za hitrost ali položaj, kar posledično predstavlja slabše rezultate pri sledenju poti. Robotska baza Mesa pa ima v nasprotju z Jaguarjem vgrajene koračne motorje z regulacijsko zanko s PID regulatorjem, kar pomeni zanesljivo doseganje podanih hitrosti in natančnejše sledenje podane poti.



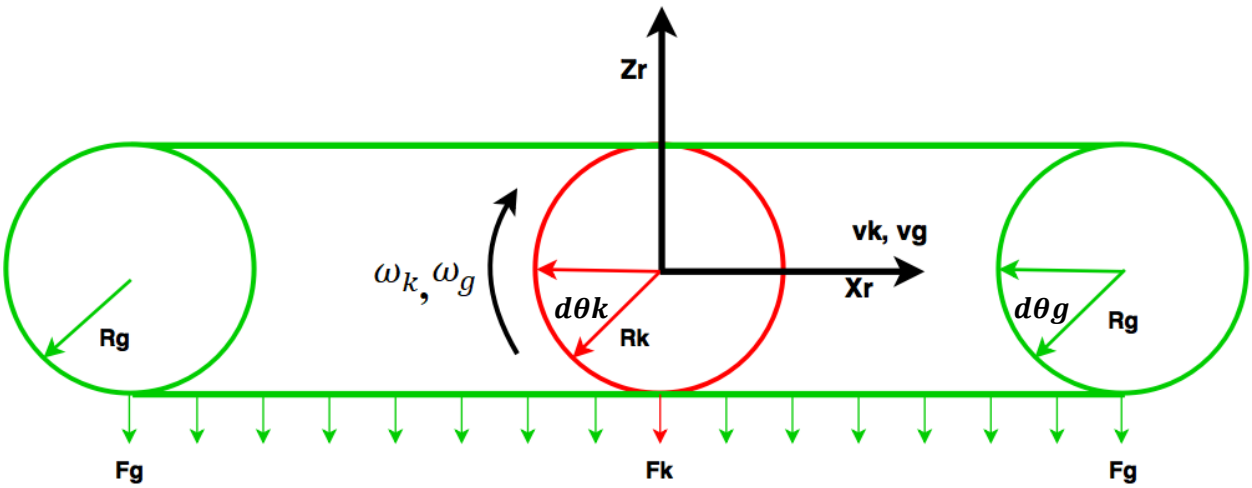
Slika 1: Model in slika mobilnega robota Jaguar.



Slika 2: Model in slika mobilnega robota Mesa.

2.2 KINEMATIČNI MODEL MOBILNEGA ROBOTA

Kinematični model kopenskih mobilnih robotov predstavimo na primeru goseničnega robota. V našem primeru smo imeli na voljo dva gosenična mobilna robota, Jaguarja in Meso, pri katerih je kinematični model podoben, le da ima mobilni robot Jaguar dograjene premične ročice, s katerimi si pomaga pri plezanju čez ovire. Opisali bomo kinematični model robota in osnovne enačbe gibanja robota v prostoru. Vse osnovne enačbe so izpeljane iz kinematičnega modela kolesa. Koordinatni sistem mobilnega robota je mogoče videti na Sliki 3, kjer osi X-Y-Z predstavlja zunanji globalni koordinatni sistem, Xr-Yr-Zr pa je lokalni robotski koordinatni sistem mobilnega robota. Kinematični model robota povzemamo iz [3], [34], [54] in [55].



Slika 3: Primerjava med kolesnim (rdeča barva) in goseničnim (zelena barva) pogonom.

V tem delu obravnavamo gosenične mobilne robote, ki so podobni kolesnim mobilnim robotom. S kinematičnim modelom kolesnega mobilnega robota si prav tako pomagamo pri izračunu kinematičnega modela goseničnega robota. V primeru kolesa v idealnih pogojih ni zdrsa in se kolo dotika tal v eni točki. Če se pojavijo zdrsi, pozicionirni sistem kolesa pridobi napačne podatke o položaju, relativno na globalni koordinatni sistem in s tem se pojavijo napake pri regulaciji in položaju mobilnega robota. Nasprotno pa gosenični mobilni robot za vrtenje nujno potrebuje zdrse, saj je površina gosenice v stiku s podlago precej večja od kolesa. Gosenični mobilni robot tako potrebuje več energije za premikanje, saj mora premagovati sile trenja med podlago in gosenico. Dobra stran gosenic je precej večji oprijem na podlagi, kar poveča možnost premagovanja ovir. Če izpostavimo primer stopnic; z enostavnim kolesnim mobilnim robotom (brez dodatnih manipulatorjev koles) je praktično nemogoče premagovati stopnice ali stopničaste ovire. Gosenični mobilni robot, ki ima večjo oprijemalno površino, pa stopnice lahko premaguje, pod pogojem da zmore premagati višino stopnice. Slaba stran goseničnih mobilnih robotov je snemanje gosenic v primeru zatika gosenice, kar je po navadi pogojeno z mehansko konstrukcijo mobilnega robota. Na kompleksnih tleh, kot so ruševine, prah, pesek, ipd., je smiselno uporabiti gosenični pogon mobilnega robota, predvsem zaradi večje oprijemalne površine. Kinematični model goseničnega mobilnega robota računamo podobno kot v primeru kolesnega mobilnega robota, kjer se upoštevajo zdrsi [3].

Kolesni pogon je na sliki 3 predstavljen z rdečo barvo, gosenični pogon pa z zeleno barvo, Rk in Rg predstavljata premer kolesa oziroma gosenice. Vse veličine so predstavljene v globalnem koordinatnem sistemu v ravnini Zr-Xr. V primeru vrtenja v izhodišču ravnine Zr-Xr, se pojavijo linearne hitrosti vk in vg in kotne hitrosti wk in wg. Na drsenje kolesa med vrtenjem vpliva površina podlage, na katero delujejo sile; v primeru kolesa je to Fk in v primeru gosenice Fg. Gosenica ima v tem idealnem primeru enakomerno porazdeljeno težo in so vse sile Fg enake. Relacije med kotno in linearno hitrostjo kolesa so opisane z enačbami (1, 2 in 3).

$$Rk = Rg \tag{1}$$

$$\omega_k = \frac{d\theta k}{dt}, \omega_g = \frac{d\theta g}{dt} \tag{2}$$

$$vk = \omega_k * Rk, \quad vg = \omega_g * Rg \tag{3}$$

Slika 3 pokaže, da sta v izbranem primeru premera kolesa in gosenice enaka. Spremembo kota kolesa in spremembo kota zasuka gosenice lahko zapišemo s spremenljivkami $d\theta k$ in $d\theta g$, kotna hitrost kolesa ali gosenice pa je definirana s spremembo kota po času, kar lahko uporabimo za izračun linearne hitrost

oziroma obodne hitrosti, izračunane s pomočjo radija kolesa Rk in Rg. Povzamemo lahko naslednje zakonitosti, ki veljajo za kolo in za gosenico (4).

$$vk = \omega_k * Rk \rightarrow Rk = \frac{vk}{\omega_k} \tag{4}$$

Kolesni ali gosenični robot ima dve kolesi ali dve gosenici, za vsako kolo ali gosenico veljajo omenjene zakonitosti, ki jih lahko pretvorimo v naslednji zapis. Če pogledamo sliko 4, vidimo dve linearni hitrosti koles vl in vr (levega in desnega kolesa), ti hitrosti predstavljata gibanje robota v globalni X-Y ravnini in ne več v Zr-Xr ravnini, kot smo ga obravnavali do sedaj. Kotni hitrosti leve in desne strani zapišemo z ω_l in ω_r, relacijo med kotno in linearno hitrostjo pa predstavljata enačbi (5) in (6).

$$vl = \omega_l * Rk \tag{5}$$

$$vr = \omega_r * Rk \tag{6}$$

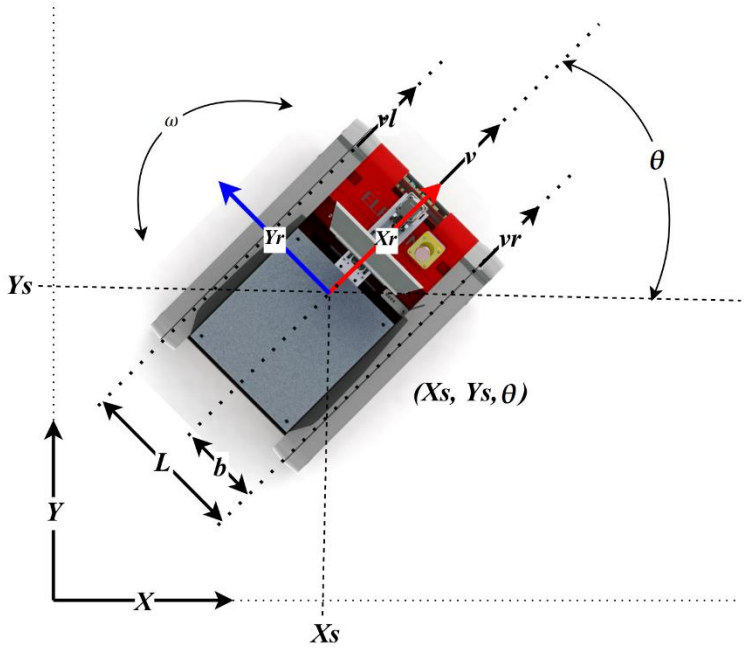
$$R = v/\omega \tag{7}$$

Med kolesi je konstantna razdalja, ki je definirana s parametrom b, ta predstavlja polovico razdalje med kolesi. V primeru enakomerne hitrosti vr in vl se robot giblje enakomerno naravnost in generira skupno linearno hitrost v. Če pa se kolesa gibajo z različnima hitrostma vr in vl, potem to povzroči zavoje. V nadaljevanju bomo podrobneje obravnavali kinematiko goseničnega mobilnega robota z upoštevanjem zdrsov pri vrtenju.

Izpeljava se prične s kinematičnimi zakoni v odvisnosti od časa, te smo že predstavili s predpostavko, da se lahko translacijska in kotna hitrost robota krmilita neodvisno z omejenimi navori. Trajektorija predstavlja krog s polmerom R (7), kjer je ω kotna hitrost in v kotna hitrost. Naj bo $x(t)$ in $y(t)$ pozicija robota v prostoru ter θ orientacija robota v prostoru, trojica $(x(t), y(t), \theta)$ predstavlja kinematično konfiguracijo robota v prostoru. Naj bo $x(t_n), y(t_n)$ in $x(t_0),\ y(t_0)$ pozicija robota v prostoru v času t_0, kjer velja:

$$x(t_n) = x(t_0) + \int_{t_0}^{t_n} v(t) * \cos\theta(t) * dt \tag{8}$$

$$y(t_n) = y(t_0) + \int_{t_0}^{t_n} v(t) * \sin\theta(t) * dt \tag{9}$$



Slika 4: Koordinatni sistemi mobilnega robota, definirani kotna in translacijska hitrost.

Enačbi sta odvisni od hitrosti robota. Zamenjamo lahko $v(t)$ in $\theta(t)$ s kinematično enačbo in dinamično konfiguracijo $v(t_0), \theta(t_0), \omega(t_0)$ ter pospeškom $v(t)$ in $\omega(t)$, dobimo:

$$x(t_n) = x(t_0) + \int_{t_0}^{t_n} \left(v(t) + \int_{t_0}^{t_n} \dot{v}(t) * dt\right) * \cos\left(\theta(t) + \int_{t_0}^{t} \left(\omega(t) + \int_{t_0}^{\hat{t}} \dot{\omega}(dt) * dt\right) * dt\right) * dt \tag{10}$$

Sedaj so enačbe v obliki, kjer so trajektorije robota odvisne od njegove začetne konfiguracije v času t_0, za katere predpostavimo, da jih lahko krmilimo. Zgornjo enačbo lahko poenostavimo tako, da je med časom t_0 in t_n lahko robot krmiljen z neskončnim številom krmilnih signalov, kjer n kaže število teh krmilnih signalov.

Za izpeljavo bolj praktičnega modela bomo poenostavili zgornjo enačbo s približkom robotove hitrosti na časovnem intervalu $[t_i, t_{i+1}]$ s konstanto.

$$x(t_n) = x(t_0) + \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} v_i * \cos(\theta(t) + \omega_i * (t - t_i)) * d\hat{t} \tag{11}$$

Če rešimo integral, dobimo:

$$x(t_n) = x(t_0) + \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} (F_x^i(t_{i+1})) \tag{12}$$

$$F_x^i(t) = \begin{cases} \frac{v_i}{\omega_i}\left(sin\theta(t_i) - \sin\left(\theta(t_i) + \omega_i * (t - t_i)\right)\right), & \omega_i \neq 0 \\ v_i \cos\left(\theta(t_i)\right) * t, & \omega_i = 0 \end{cases} \quad (13)$$

Podobno velja tudi za Y koordinato. Opazimo, če je $w = 0$, se bo robot gibal po ravni črti, in če $w \neq 0$, se bo gibal po krožni trajektoriji, kjer:

$$M_x^i = -\frac{v_i}{w_i} * \sin\theta(t_i) \quad (14)$$

$$M_y^i = -\frac{v_i}{\omega_i} * \cos\theta(t_i), \text{ kjer velja:} \quad (15)$$

$$\left(F_x^i - M_x^i\right)^2 + \left(F_y^i - M_y^i\right)^2 = \left(\frac{v_i}{\omega_i}\right)^2 \quad (16)$$

Predstavljene enačbe so odvisne samo od hitrosti, ki jih določajo polmeri trajektorij. Primer trajektorij za našega robota je prikazan na sliki 5, kjer črna črta prikazuje trajektorijo robota glede na kotno in translacijsko hitrost v X-Y prostoru s krožnico s polmerom R. Do sedaj smo razložili, kako robota voditi glede na sinhrone ukaze translacijske in kotne hitrosti, sedaj pa poglejmo podrobnosti modela našega robota.

Kinematični model v univerzalnem koordinatnem sistemu je definiran kot:

$$v = \frac{vl+vr}{2}, \omega = \frac{vl-vr}{2b} \quad (17)$$

Hitrost na levi in desni gosenici, v odvisnosti od skupne hitrosti, je definirana kot:

$$vl = v - \frac{\omega*2*b}{2} = v - \omega * b \quad (18)$$

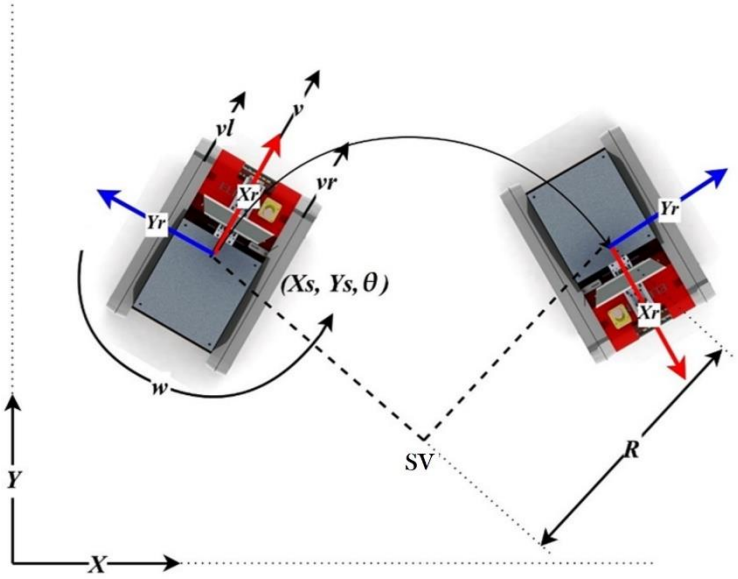
$$vr = v + \frac{\omega*2*b}{2} = v + \omega * b \quad (19)$$

v in ω sta translacijska in kotna hitrost robota, R je premer pogonskega kolesa, ω_r in ω_l sta kotni hitrost, i_r in i_l sta drsna koeficienta gosenice ter $L = 2b$, c je enako konstanti $\frac{r}{2L}$,

$$\frac{d}{dt}\begin{bmatrix} X \\ Y \\ \varphi \end{bmatrix} = \begin{bmatrix} \frac{R}{2}\overline{i_r}\cos(\varphi) & \frac{R}{2}\overline{i_l}\cos(\varphi) \\ \frac{R}{2}\overline{i_r}\sin(\varphi) & \frac{R}{2}\overline{i_l}\sin(\varphi) \\ c\overline{i_r} & -c\overline{i_r} \end{bmatrix}\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \quad (20)$$

Zgornja enačba predstavlja kinematični model avtonomnega mobilnega robota, ki na podlagi kotnih hitrosti modelira skupno hitrost, ta predstavlja srednjo vrednost kotnih hitrostih, ter kot zasuka glede na razliko kotnih hitrostih. Poenostavljen kinematični model robotske baze, kjer Xs, Ys, θ predstavljajo

položaj in orientacijo robota v globalnem koordinatnem sistemu X-Y, lahko zapišemo v odvisnosti položaja od translacijske in kotne hitrosti mobilnega robota (21).



Slika 5: Krožno gibanje kopenskega mobilnega robota z diferencialnim pogonom.

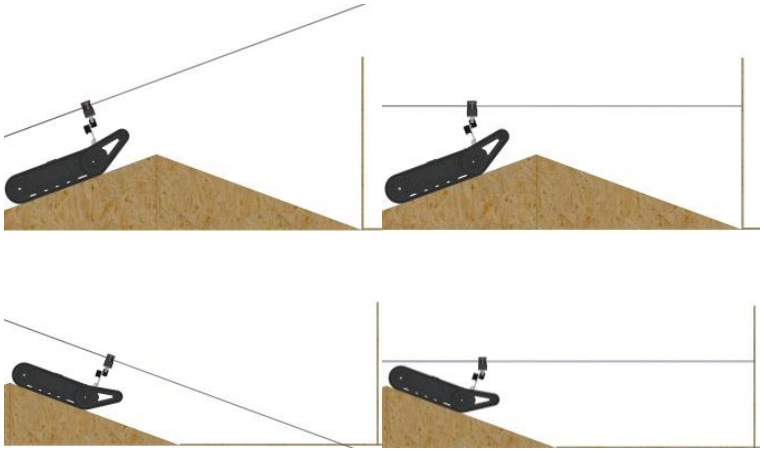
$$\begin{bmatrix} Xs \\ Ys \\ \theta \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{21}$$

Slika 5 predstavlja gibanje robota po ravnini X-Y, kjer lahko vidimo središče vrtenja (SV); ta točka predstavlja središče, okoli katerega se robot v nekem trenutku vrti. Gibanje robota po različnih podlagah vpliva na različne oprijeme gosenic, kar vpliva na položaj SV. Če se položaj SV spremeni, potem robot ne izdela pravilne oblike zavoja, trajektorija robota ni idealen krog ali polkrog. To premikanje SV imenujemo zabrisanost centra rotacije. Zabrisanost poleg nepredvidljivih oprijemov gosenic na podlagi povzroča tudi neenakomerna porazdelitev teže na gosenice. Neenakomerna porazdelitev teže na gosenice vpliva na zamik SV iz težiča robota. Uporabljeni roboti so gnani z gosenicami, kar omogoča robotu za reševanje veliko stopnjo mobilnosti, to področje kinematike mobilnega robota na diferencialni pogon smo obravnavali v [5].

2.3 SISTEM ZA IZRAVNAVANJE LASERSKEGA MERILNIKA RAZDALJE

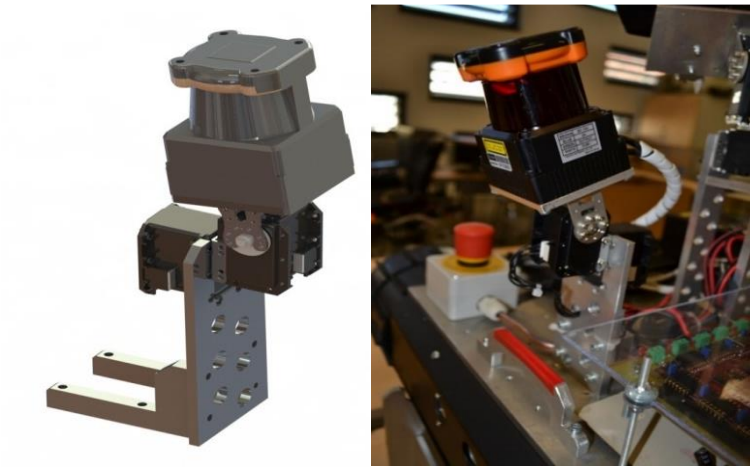
Avtonomni mobilni robot se lahko giblje po zelo razgibani podlagi. Problem je predstavljen na sliki 6, kjer na sliki levo lahko vidimo, kako nepremični laser vpliva na podatke laserskega merilnika razdalje, in rešitev

na sliki na desni strani, kjer se s pomočjo sistema za uravnavanje prilagaja položaj LMR. Problem in rešitev, povezana z nepravilno gradnjo zemljevida, kar smo izpostavili v hipotezi **H2**.



Slika 6: Levo mobilna baza z nepremično nameščenim LMR in desno mobilna baza s sistemom za uravnavanje LMR.

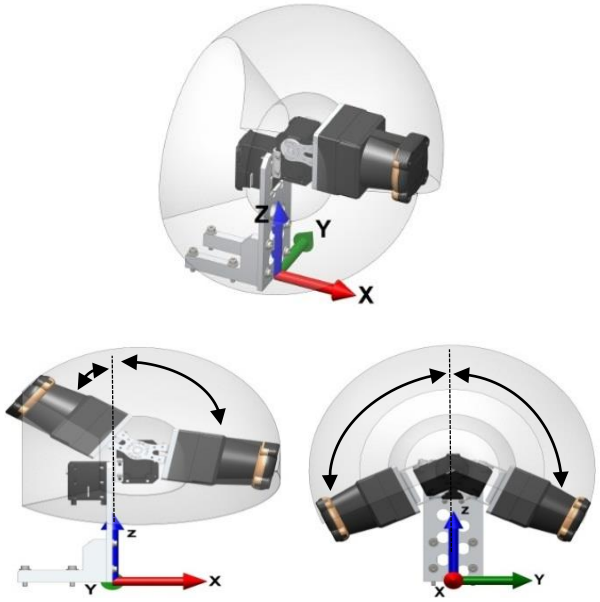
Naloga sistema za uravnavo LMR je, da vedno drži LMR v X-Y ravnini, ne glede na naklon mobilne platforme. Prednost tega je, da imamo vedno realno sliko okolice, ki nam jo LMR poda v ravnini X-Y, ki je premaknjena od tal za določeno razdaljo. Tako dobimo prave podatke o lokaciji sten in ovir na poti robota.



Slika 7: Sistem za uravnavo LMR (model na levi in na desni realiziran).

Za zajemanje meritev oddaljenosti ovir od robota je uporabljen LMR Hokuyo UTM-30LX, ki ima 0,25° kotno resolucijo v območju 270° z možnostjo meritev razdalje do 30 m. Natančnost LMR je +/- 3 mm. Za premikanje se uporabljajo kvalitetni servomotorji proizvajalca Robotis. Komunikacija med računalnikom in servomotorji poteka preko RS485 protokola, za katerega potrebujemo RS485 na USB pretvorniku signala. Za zaznavanje naklonov mobilne baze skrbi senzor IME (inercijska merilna enota), ki poda zasuke

glede na osi v lastnem koordinatnem sistemu senzorja. Položaj senzorja je treba upoštevati pri izračunih pravih zasukov glede na geometrijski center mobilne baze robota.



Slika 8: Zgoraj delovni prostor sistema za uravnavanje LMR, uravnavanje naklona glede na os Y; –55/+100° (spodaj levo), in X os +/-115° (spodaj desno).

Teoretični dokaz smo izvedli na modelu mobilnega robota, predstavljenega z enačbo 22.

$$RB = R[z, \theta].R[y, \beta].R[x, \alpha] \tag{22}$$

V enačbi $R[z, \theta]$ predstavlja rotacijsko matriko, prvi parameter označuje os, nad katero se rotacija izvaja, drugi parameter pa kot zasuka na tej osi. Celotno kompenzacijo kotnih zasukov lahko zapišemo z enačbo 23.

$$I = RB.RB^{-1} = RB.\{R[z, \theta].RL\}^{-1}$$

$$= RB.RL^{-1}.R^{-1}[z, \theta] \tag{23}$$

I je enotna matrika, RL predstavlja model mehanizma za uravnavanje LMR in RL^{-1} je njegov inverzni model. Inverzni model RB^{-1} je množen z RB, ker je sistem za uravnavanje LMR nameščen na robotsko bazo in lahko kompenzira orientacijo LMR relativno na robotski koordinatni sistem. Rešitev za RL in RL^{-1} predstavlja enačba 25.

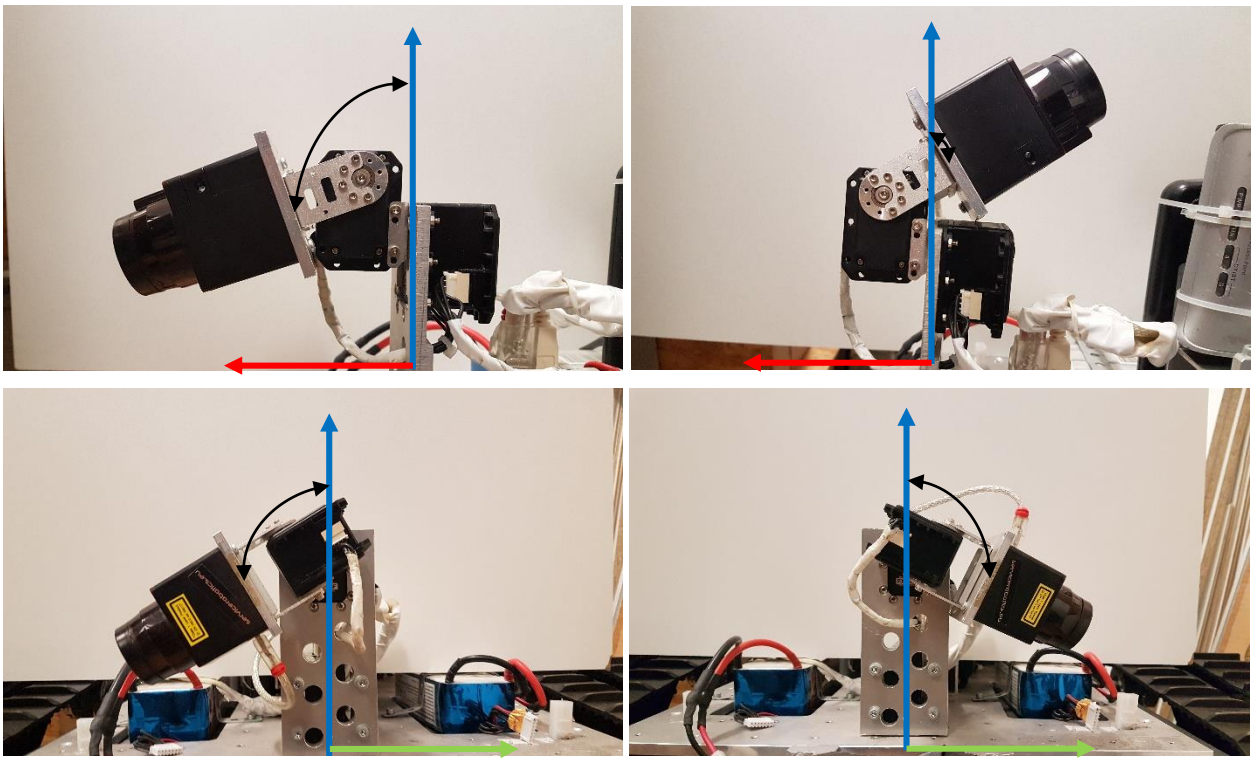
$$RL = R[y, \beta].R[x, \alpha] \tag{24}$$

$$RL^{-1} = R^{-1}[x, \alpha].R^{-1}[y, \beta] = R[x, -\alpha].R[y, -\beta] \tag{25}$$

Zasukov glede na Z os ne moremo kompenzirati s sistemom za uravnavanje LMR, zato lahko enačbo 25 zapišemo kot:

$$R[z,\theta] = \{R[z,\theta].R[y,\beta].R[x,\alpha]\}.R[x,-\alpha].R[y,-\beta] \tag{26}$$

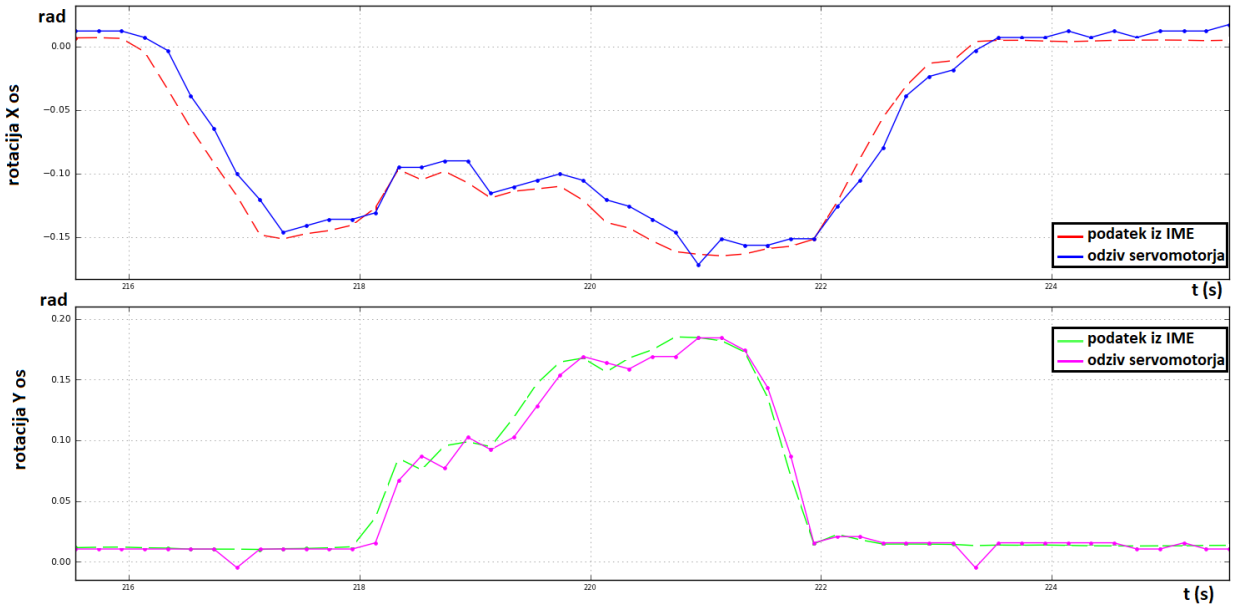
Enačba 26 dokazuje, da lahko s sistemom za uravnavanje LMR uravnavamo naklone v dveh oseh glede na zasuke robotske baze.



Slika 9: Prikaz zgrajenega sistema, zgoraj prikaz končnih položajev sistema za uravnavanje LMR glede na os Y (zelena) in spodaj prikaz končnih položajev glede na os X (rdeča).

Na sliki 10 lahko vidimo časovne odzive sistema za uravnavanje LMR, kjer podatke iz senzorja IME predstavlja prekinjena črta (rdeča, zelena), odzivi servomotorjev pa so prikazani z neprekinjeno črto (modra, vijolična), za X na zgornjem in Y na spodnjem grafu. Vrednosti kotnih premikov so predstavljene v radianih. Opazimo lahko, da se nenadni premiki odražajo v mili sekundnih zamikih, kar je več kot dovolj da pridobimo podatke iz LMR, kateri delujejo v večini primerih z 10 Hz. Sistem za uravnavanje LMR je bil razvit z namenom izdelave kvalitetnejšega kartiranja. Sistem smo testirali na številnih testnih vožnjah in tekmovanjih »RoboCup reševanje«. Avtonomni mobilni roboti morajo potovati po zelo razgibani podlagi, kot so na primer neurejene površine po potresu. Takšne razgibane podlage se na tekmovanjih »RoboCup reševanje« sestavijo z namenom poustvaritve realnih pogojev. Slika 9 prikazuje zgrajen elektro-mehanski

sistem, kjer smo s pomočjo kotomerov izmerili kotne razlike, katere so se pojavile pri premikih servo motorjev.

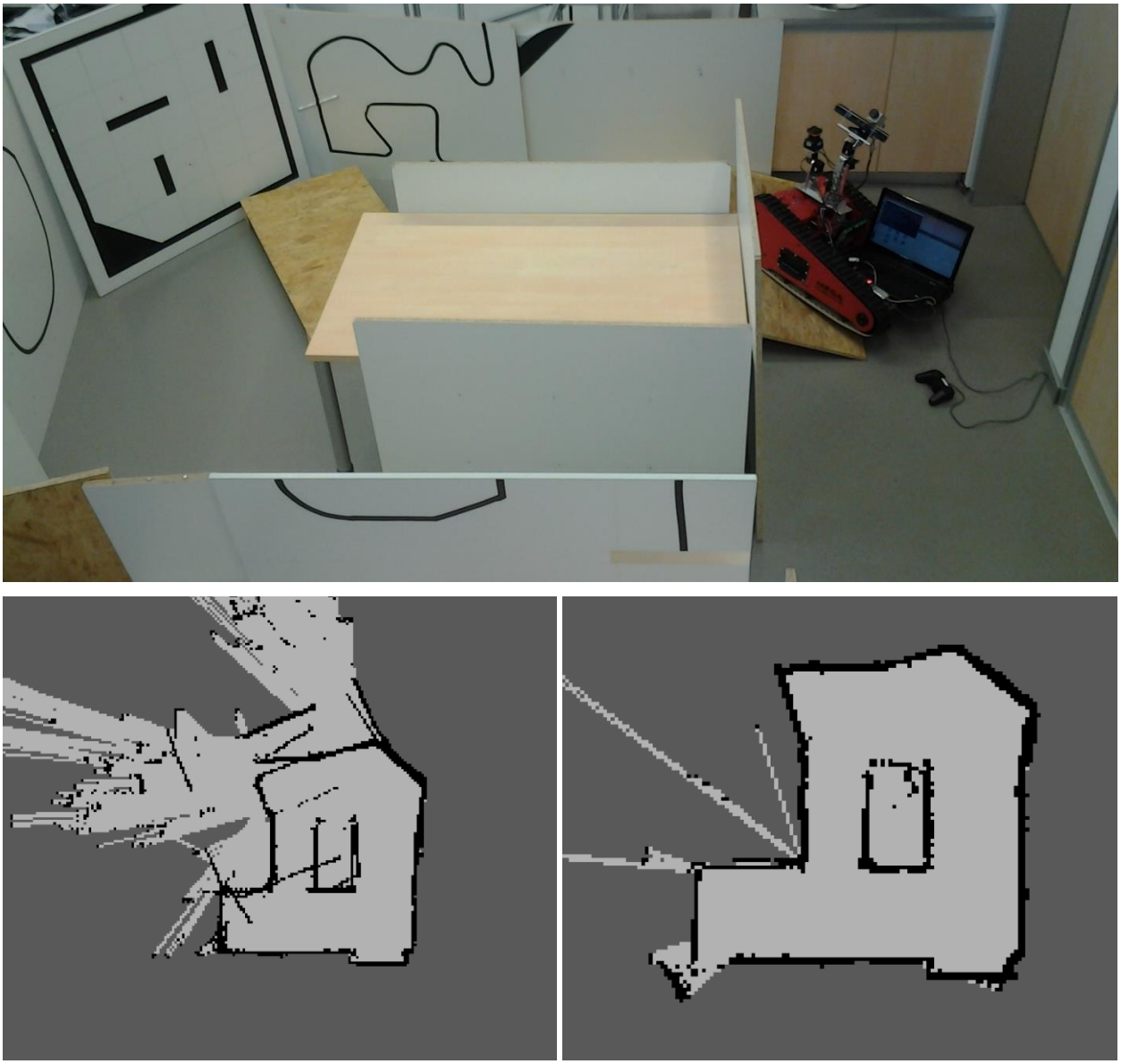


Slika 10: Odzivi sistema za uravnavanje LMR glede na zasuke po X in Y osi v odvisnosti od časa.

Za rešitev problema boljše natančnosti izdelanega zemljevida smo predlagali sistem, ki sestoji iz dveh pametnih servomotorjev in mehanskih elementov, ki omogočajo uravnavanje LMR. Slednje je omogočeno glede na Y os od -10° do +185° in glede na X os od + 115° do -115°. Prav tako smo analizirali odzivnost sistema na zasuke mobilne baze glede na IME. Ta je pritrjen neposredno na mobilno bazo, in v primeru zasuka mobilne baze poda kote trenutnega zasuka. Na podlagi podanih kotov izračunamo transformacijo tako, da je LMR vedno v vodoravnem položaju. Sistem za uravnavanje LMR smo podrobneje opisali v članku [6].

V laboratoriju smo izdelali testno reševalno okolje prikazano na sliki 11. V tej areni so se nahajale dve oviri prva klančina s 22.5 stopinjskim naklonom in ena stopnica v višini 10 cm. Najprej smo v tej areni izdelali zemljevid, kjer smo izključili sistem za uravnavanje LMR, rezultat je prikazan na sliki 11 levo. Vidimo lahko, da je zemljevid precej nerealen, saj se pojavijo navidezne stene in prav tako se pojavijo napake v okolici, kar je posledica SLAM algoritma, kateri skuša dobljene (nepravilne) podatke o okolici umestiti v obstoječ zemljevid. Posledica nepravilnih podatkov iz LMR razdalje se prikaže v obliki nepravilnih ali dodatnih sten. Na sliki 11 desno pa lahko vidimo primer, kjer smo na istem testnem okolju izvedli eksperiment s uporabo sistema za uravnavanje LMR. Opazimo lahko očitno izboljšanje brez nepravilnih sten in pravilno zarisane obstoječe stene. Na podlagi teh rezultatov lahko sklepamo, da je mobilni robot s uporabo sistema za

uravnavanje LMR veliko bolj zanesljiv, saj izdela pravilen zemljevid okolice, kar predstavlja osnovo za vse nadaljnje avtonomne operacije.



Slika 11: Zgoraj posnetek testno izdelanega okolja v laboratoriju s ovirami, spodaj levo izdelan zemljevid brez uporabe in spodaj desno s uporabo sistema za uravnavnje LMR.

Glede na rezultate raziskave, predstavljene na sliki 10, in realnih testiranj, lahko potrdimo hipotezo H2. Predvidevali smo namreč sistem, ki bo v realnem času uravnaval LMR. Rezultati predstavljajo do 100 milisekundne zamike, kar lahko v našem primeru obravnavamo kot odziv v realnem času, saj pravočasno uravna LMR in s tem ne izriše neobstoječih sten. Drugi del H2 predpostavlja da bo sistem za uravnavanje LMR ter pri gibanju naprej in nazaj lahko kompenziral naklone od -45° do +45°, v razponu 90°. Tudi to je

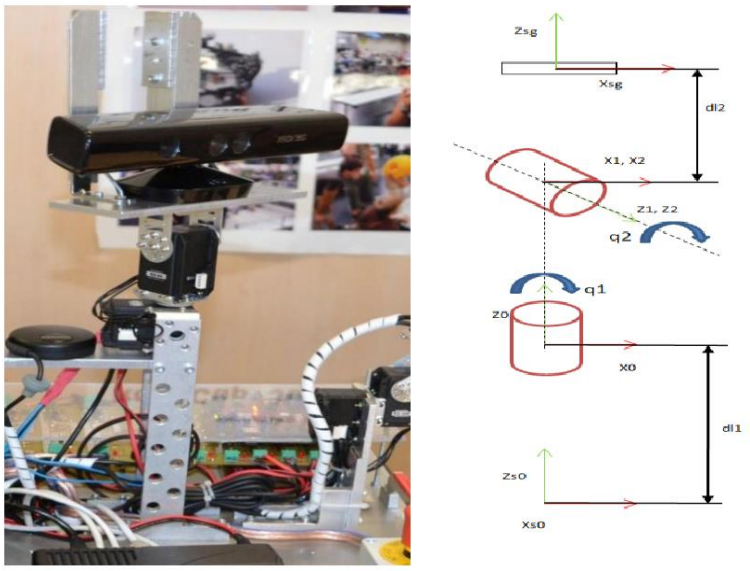
bilo uspešno izvedeno tako za os X (od + 115°do -115°), kjer smo dosegli omogočeno poravnavo v razponu 230°, kakor tudi za os Y, kjer smo dosegli od - 10° do +185°, v razponu 195°. S tem je omogočeno ustvarjanje bolj natančnega zemljevida okolice med gibanjem robota v neurejenih okoljih.

2.4 SISTEM ZA ISKANJE ŽRTEV

Težave pri prepoznavanju oseb se pojavijo pri iskanju ljudi v nepredvidljivih in nestrukturiranih urbanih okoljih, kjer lahko prepoznavo moti veliko dejavnikov, npr. prah, temperatura, ovire … V tem delu smo skušali razviti multi-senzorski sistem, ki bo zaznaval ljudi v različnih položajih. Predpostavimo, da je multi-senzorski sistem nameščen na mobilnem robotu, ki se giblje v nekontroliranih okoljih, kot so urbana področja ali v naravi. Gosenična izvedba mu omogoča premikanje po zahtevnem terenu. Med potovanjem robot izdeluje načrt okolice in avtonomno prepoznava žrtve – ljudi, saj so na njem nameščeni sistemi za kartiranje, iskanje žrtev in sistem za diagnostiko. V tem podpoglavju se bomo osredinili na sistem za iskanje žrtev (na kratko SIŽ), ki ga sestavljajo mehanski nosilci s servomotorji in senzorji.

Ideja za razvoj takšnega algoritma izhaja iz predpostavke, da mora biti prepoznava ljudi avtomatska, saj komunikacija z mobilnim robotom ni vedno zagotovljena. Namen reševalnega mobilnega robota je avtonomno preiskovanje v prostorih in stavbah, kjer brezžično omrežje ni vedno na voljo. Tudi če mobilni robot izgubi komunikacijo z operacijsko postajo, mora še naprej izvajati svojo nalogo raziskovanja in zanesljivega iskanja žrtev. Robot javi lokacije prepoznanih žrtev oziroma ljudi, zabeleženih na izdelanem zemljevidu, takoj ko je to mogoče. Prve verzije algoritma za prepoznavo smo testirali v kontroliranem okolju – v laboratoriju. Naslednji testi algoritma za prepoznavo pa so bili izvedeni v simulirani areni potresa, ki je bila standardizirana s strani agencije NIST (National Institute of Standards and Technology) [7].

Na sliki 12 (levo) je vidno, da je na senzorski glavi nameščen senzor Kinect in termalna kamera. S temi senzorji optično preberemo okolico in iščemo žrtve, ki so lahko simulirane z različnimi toplotnimi izvori, oblikami in oznakami. Mehanski sistem z motorji služi za premikanje senzorskih delovnih prostorov v želene smeri, kjer se iščejo potencialne žrtve. Sistem je zgrajen iz dveh servomotorjev Robotis Dynamixel RX-64. Prvi servomotor vrti senzorsko glavo glede na Z os robota, drugi servomotor pa vrti senzorsko glavo glede na Y os robota. Glede na tipe senzorjev, ki imajo širok spekter zaznave, tako pokrijemo velik del prostora, kjer se lahko nahajajo žrtve. Na sliki 12 (desno) je mogoče videti model sistema senzorske glave. S homogenimi transformacijami lahko izpeljemo kinematični model sistema senzorske glave.



Slika 12: Sistem senzorske glave in desno skica kinematičnega modela sistema senzorske glave.

$$\mathbf{A} = \begin{bmatrix} c(q1)c(q2) & -s(q1) & -c(q1)s(q2) & -dl2c(q1)s(q2) \\ s(q1)c(q2) & c(q1) & -s(q1)s(q2) & -dl2s(q1)s(q2) \\ s(q2) & 0 & c(q2) & dl2c(q2) + dl1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{27}$$

Enačba 27 prikazuje izpeljan kinematični model senzorske glave s pomočjo Denavit – Hartenberg postopka, kjer $q1$ in $q2$ predstavljata zasuk po oseh servomotorja 1 in 2. Omenjeni sistem smo poimenovali multi-senzorski, ker ga sestavljata dva senzorja: termalni senzor Flir PathFinder in Xboxov igralni senzor Kinect. Termalna kamera nam nudi en podatkovni tok, medtem ko senzor Kinect pošilja poleg slikovnega toka še oblak točk in podatke o zvoku. Oba senzorja sta nameščena na vrh drugega servomotorja, prikazano na sliki 12. Senzorja sta usmerjena v isto smer, vendar nimata iste resolucije in vidnega kota, zato je potrebna poravnava slik.

Termalna kamera Flir PathFinder zajema slike z resolucijo 320 × 240 slikovnih točk z merilnim območjem od -40 do +80 °C. Senzor nam vrne sivinsko sliko s približno temperaturno resolucijo 0,47°. Termalna kamera ja v osnovi namenjena za potrebe avtomobilske industrije. Kamera ima precej majhen vidni kot, razpona 36°, je vodoodporna in tehta samo 360 g. Izhod iz kamere je analogni signal, ki ga preko zajemalnika pretvorimo v digitalni signal. Uporabili smo tudi **Xboxov zmogljiv igralni senzor Kinect**, ki ima integrirano RGB kamero, globinsko enoto, ki jo sestavlja infrardeči projektor, črno-beli senzor in večnamenski mikrofon. S tem zmore popolno 3D-identifikacijo, prepoznava obrazne poteze in glas uporabnika. Senzor Kinect uporabljamo za pridobitev RGB slike ter oblaka točk. Kinect nam vrne sliko v velikosti 640 × 480 slikovnih točk, in prav tako isto število točk v prostoru; če je te mogoče pridobiti.

Tovarniške specifikacije omogočajo delovanje med 1,5 m in 3,5 m oddaljenosti, vendar nam goli podatki podajajo razdalje med 0,6 m in 6 m. Ideja Kinecta je bila ustvariti naravni uporabniški vmesnik, ki predstavlja skupek tehnologij in omogoča, da robota vodimo brez fizičnega upravljalnika. Edini naravni upravljalnik je naše telo. Bistvo tehnologije je, da preko senzorjev, kamer in mikrofonov zazna gibe telesa oz. njegovih delov v realnem času. Ta pa sloni na knjižnicah za obdelavo vhodnih podatkov. Imamo senzor, ki preko naravnega uporabniškega vmesnika zaznava in pošilja: slikovni tok, globinski tok in zvočni tok podatkov. Senzor globine je sestavljen iz dveh delov: projektorja IR svetlobe in senzorja IR svetlobe. Povezana sta v stereo način, to pomeni, da simulirata način daljnogleda oz. človeškega vida, ki se uporablja za izdelavo 3D-slik. Literaturo za obdelavo slik smo črpali iz literature v [8–14].

Za programiranje sistema za iskanje žrtev smo uporabili ROS in OpenCV knjižnico za obdelavo slik. Prepoznavo ljudi smo izvajali s spodaj opisanimi postopki.

- **Toplotna prepoznava**

  Iz sivinske slike smo določili, kateri sivinski odtenek predstavlja normalno temperaturo človeka 37°C (150–160 binarna vrednost). Nato smo nižje vrednosti temperature primerno skalirali do 0, prav tako smo do 0 sklairali vrednosti nad 37° (> 160). Na tak način se nam je na verjetnosti sliki največja vrednost pojavila prav na mestih s temperaturo 37°. Na sliki 13 (skrajno desno zgoraj) vidimo prepoznavo zaznane temperature človeka.

- **Prepoznava robov**

  Nad vhodno sliko smo izvedeli zaznavo robov s Sobel operatorjem, ki nam je vrnil sliko z robovi, ki so na njej predstavljeni kot sivinske vrednosti. Nad isto vhodno sliko smo izvedli Cannyjev detektor robov, ki je vrnil binarno sliko. Nad to sliko smo nato izvedli Hougovo transformacijo, kjer smo izločili ravne črte. To smo naredili z namenom izločitve okolja iz prepoznanih robov, tako smo dosegli, da je na sliki ostala le silhueta oziroma obrisi neravnih elementov, ki potencialno lahko predstavljajo ljudi. V sliko, na kateri je bila izvedena Hougova transformacija, smo v nadaljevanju preslikali vrednosti, ki so se ujemale z Sobelovovim detektorjem robov. Tako smo dobili verjetnostno sliko z verjetnostnimi podatki samo na prepoznanih robovih.

- **Zaznava gibanja**

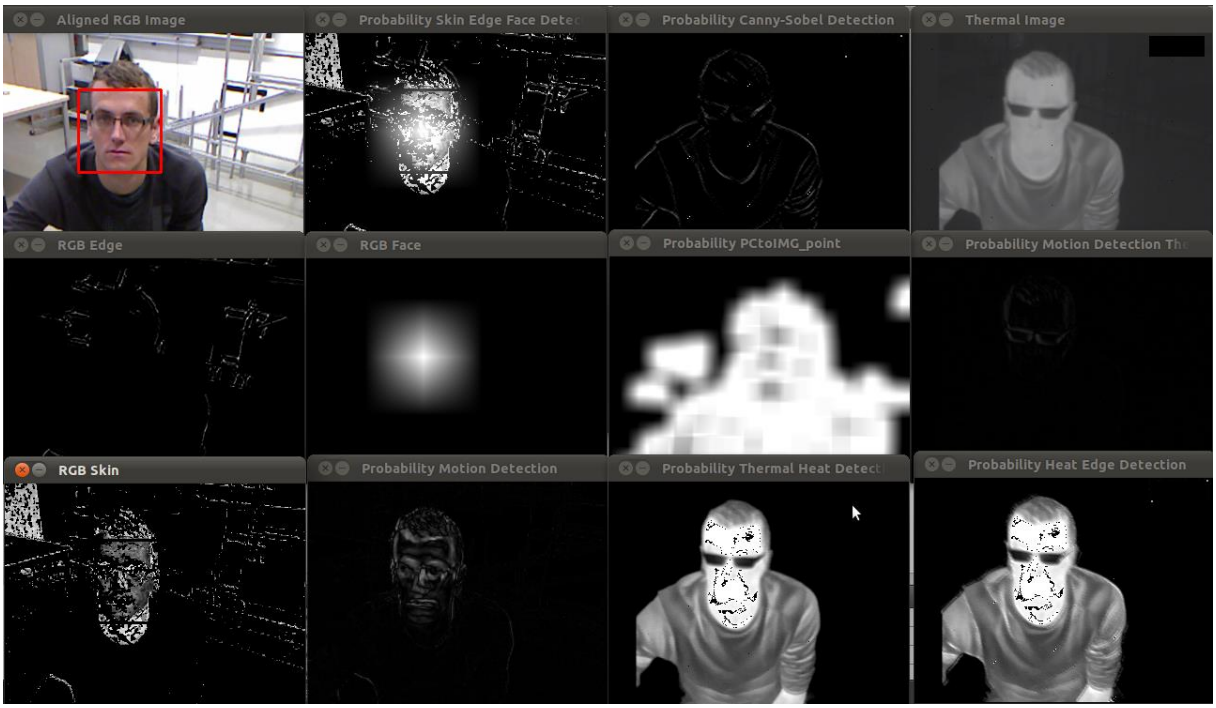
  Gibanje smo zaznavali z enostavnim odštevanjem slik, kje smo zaporedni sliki odšteli, nad rezultatom pa izvedli pragovno segmentacijo zaradi premikanja senzorske glave.

- **Prepoznava kože**

  Vhodno RGB sliko smo pretvorili v YUV format in nato naredili pragovno metodo z določenim pasom barvnih vrednosti, ki so ustrezali barvi kože. Sredina v teh pasovih je predstavljala največjo verjetnost, da se na dani slikovni točki nahaja koža, ostale vrednosti so skalirane proti nič. Tako smo dobili dve verjetnostni sliki z dvema različnima postopkoma. Verjetnostni sliki smo nato povprečili in dobili izhodno sliko verjetnosti za prepoznavo kože.

- **Prepoznava obraza**

  Za prepoznavo obraza smo uporabili Haar detektor, pri katerem je obraz opisan z znanilkami v html datoteki. Algoritem poišče na vhodni RGB sliki lokacije obraza, kjer je izhodni podatek okvir. Verjetnost, da se v tem okvirju nahaja obraz, smo razporedili tako, da je sredina okvirja predstavljala največjo verjetnosti, ki se je manjšala proti nič v določenem polmeru, določenem z velikostjo vrinjenega okvirja iz detektorja Haar.



Slika 13: Vhodni sliki za zaznavo verjetnosti vse rezultirajoče vrjetnostne slike.

- **Prepoznava oblike**

  Oblak točk, pridobljen iz senzorja Kinect, smo najprej filtrirali s kocko velikosti 5 cm^3, z namenom zmanjšanja števila točk v oblaku, kar je pripomoglo k hitrejšemu obdelovanju oblaka. Če se je v

kocki velikosti 5cm^3 nahajalo več točk, je rezultat predstavljal povprečje vseh točk. Po filtraciji na oblaku smo prepoznali ravnine, kjer so lahko točke izstopale v določenem pragu. Izhodni oblak je predstavljal nelinearne oblike, kjer so ravnine izločene. Izhodni oblak smo nato preslikali na 2D-ravnino, z namenom omogočanja primerjave rezultatov z ostalimi dvodimenzionalnimi senzorji (RGB in termalna kamera). Izhodna slika je predstavljala binarno sliko, ki smo jo zameglili in tako dobili večje verjetnosti tam, kjer je bil oblak najbolj intenziven.

Združevanje verjetnosti smo opravili s sivinskimi slikami, te predstavljajo vrednosti med 0 in 255. V območjih z največ prepoznanih svetlih točk slike smo lahko sklepali o prisotnosti človeka. Kot najbolj zanesljiv indikator prisotnosti človeškega telesa se je pokazala detekcija temperature in detekcija gibanja. Iz trenutnih rezultatov smo zaključili, da verjetnostne slike iz zaznave robov, zaznav kože in obraza v manjši meri pripomorejo k detekciji ljudi. Kljub vsemu, pa so nam posamezne verjetnosti iz senzorjev že nazorno prikazovale položaj človeka, zato smo omenjene verjetnosti združili. Z združitvijo imamo manj podatkov za nadaljnjo obdelavo, v kateri smo skušali z različnimi postopki izluščiti informacije o prisotnosti človeka.

2.5 DIAGNOSTIČNA PLOŠČA

Tematiko diagnostične plošče za mobilne robote je podrobneje obravnavana v literaturi [15–18]. Ideja diagnostične plošče za mobilnega robota izhaja iz dejstva, da veliko število različnih avtonomnih sistemov, vgrajenih v mobilnega robota, lahko privede do napak v sistemu. Napake so lahko programske ali mehanske, v obeh primerih pa lahko diagnostična plošča igra pomembno vlogo v avtonomiji mobilnega robota. Zagotavljanje zanesljivosti delovanja mobilnega robota smo izpostavili v hipotezi **H3**. Zadali smo si cilj izdelave elektronskega vezja, ki bi omogočalo spremljanje in interakcijo s sistemi mobilnega robota.

Avtonomni mobilni reševalni roboti za svoje delovanje potrebujejo večje število kompleksnih elektromehanskih sistemov kot so:

1. baza mobilnega robota (pogonski motorji s krmilnikom)
2. senzorski sistem za navigacijo
    a. servo motorji
    b. LMR
3. senzorski sistem za prepoznavo žrtev
    a. servo motorji
    b. kamera

c. kinect
d. drugi senzorji)

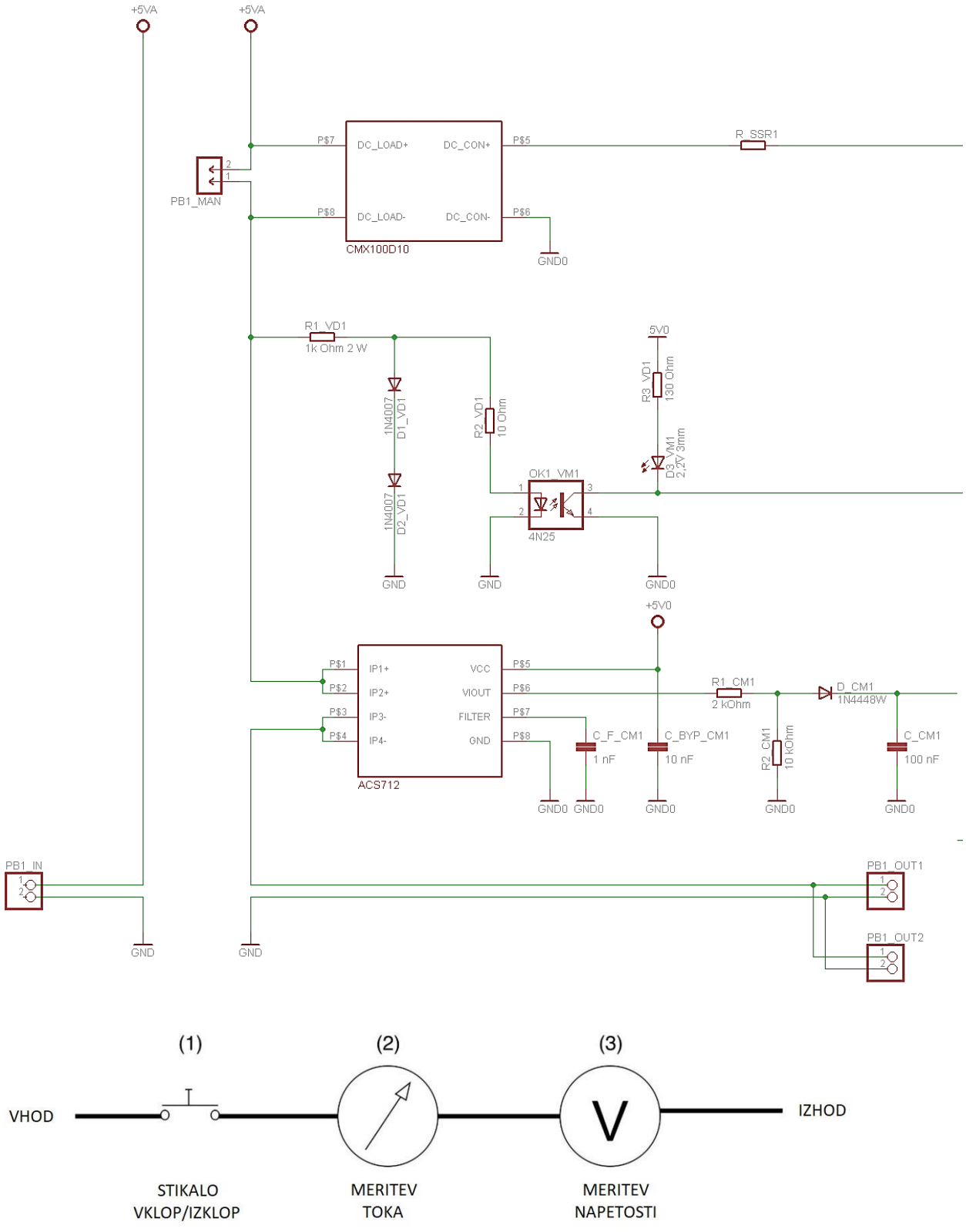
4. glavni računalnik
5. komunikacijsko stikališče (usmernik).

Vsak izmed zgoraj omenjenih sistemov potrebuje napajalno linijo, katere imajo različne zahteve. V našem primeru smo se odločili za izdelavo dveh tipov:

a. 3 x močnejša linija (maksimalno 60V in 30A)
b. 7 x šibkejša linija (maksimalno 20V in 5A).

Močnejše linje smo uporabili za večje porabnike kot so pogonski elektromotorji in procesorski sistemi, šibkejše linije pa za senzorje in servo motorje. Določili smo elektronske komponente kateri so bile primerne močnostnim zahtevam mobilnega robota namenjenega za reševanje. Uporabiti smo morali tudi krmilnik, kateri bi omogočal zajemanje podatkov in vzpostavitev komunikacije z ROS. Na krmilniku se je implementiral namenski protokol, kateri je pošiljal obstoječe informacije naprej v diagnostični sistem.

Diagnostično ploščo krmili mikrokrmilnik (PIC32-MX795F512), ki ima možnost vklopa in izklopa vsakega posameznega kanala in tudi glavne komunikacijske linije do glavnega računalnika s mrežno povezavo. Izbrali smo 80 MHz, 32-bitni mikrokrmilnik, saj zagotavlja zadostno število 85 vhodno/izhodnih signalov. Vgrajen ima 10/100 Ethernet MAC z Mii/RMII vmesniki in ima 512 K Flash in 128 K RAM pomnilnika. Za merjenje toka in napetosti je zelo pomembno, da ima dober AD pretvornik; v tem primeru imamo hitri in natančni 16-kanalni 10-bitni pretvornik. Na osnovi TCP/IP protokola je diagnostična plošča povezana z glavnim krmilnim sistemom za nadzor porabe energije sistema in izvedbo ukrepov. Mikrokrmilnik se uporablja za merjenje parametrov kanalov in kot nadzor vklopa/izklopa posameznih kanalov. Glavna procesna enota je mikrokrmilnik, ki predstavlja osrednji nadzor preko napajalnih linij. Za lažjo izvedbo smo uporabili eksperimentalno ploščo Digilent Cerebot 32MX7 z vgrajenim priključkom RJ45 za ethernet komunikacijo. Za komunikacijo na fizičnem nivoju skrbi SMSC LAN8720, ki določa standardni ethernet vmesnik. Programska oprema je zasnovana tako, da odpre strežnik, kjer lahko kdor koli v sistemu zaprosi za podatke. Krmilnik Digilent Cerebot 32MX7, digitalni in analogni priključki so povezani s posameznimi merilnimi enotami. Povezave do kanalov so galvansko izolirane. Diagnostična plošča lahko znova zažene celoten sistem in deluje kot glavni nadzorni sistem, ki se lahko aktivira v različnih scenarijih popravila. Ob zagonu sta vklopljena kanal za PC in kanal za usmerjevalnik, kasneje pa so vklopljeni vsi drugi kanali, ki jih določi višji nivojski sistem.



Slika 14: Shema enega kanala diagnostične plošče za mobilnega robota (zgoraj), grafični prikaz delovanja posameznega kanala (spodaj).

Diagnostični sistem je mogoče prilagoditi potrebam robota in uporabnika. Obstoječa implementacija diagnostične plošče ima 10 kanalov, od tega so 3 močnejši, z maksimalno dovoljeno napetostjo 60 VDC in maksimalnim tokom 15A na posamezni kanal. Šibkih je 7 kanalov, ki so omejeni z maksimalno 20 VDC in 5A. Na vsak omenjeni kanal je priključen DC-CD pretvornik, ki je odvisen od zahtev sistema, na katerega so priključeni sistemi robota. Seznam uporabljenih elementov je mogoče videti v Tabeli 1, električno shemo pa na Sliki 14. Naslednje elektronske elemente smo uporabili za:

1. vklop/izklop stikalo, rele (SSR-CMX100D10 in PVN012),
2. senzor (4N25) za detekcijo napetosti med 3,3 – 60 V in
3. Hallov tokovni senzor (ACS712) za meritev toka med 0 – 30 amperov.

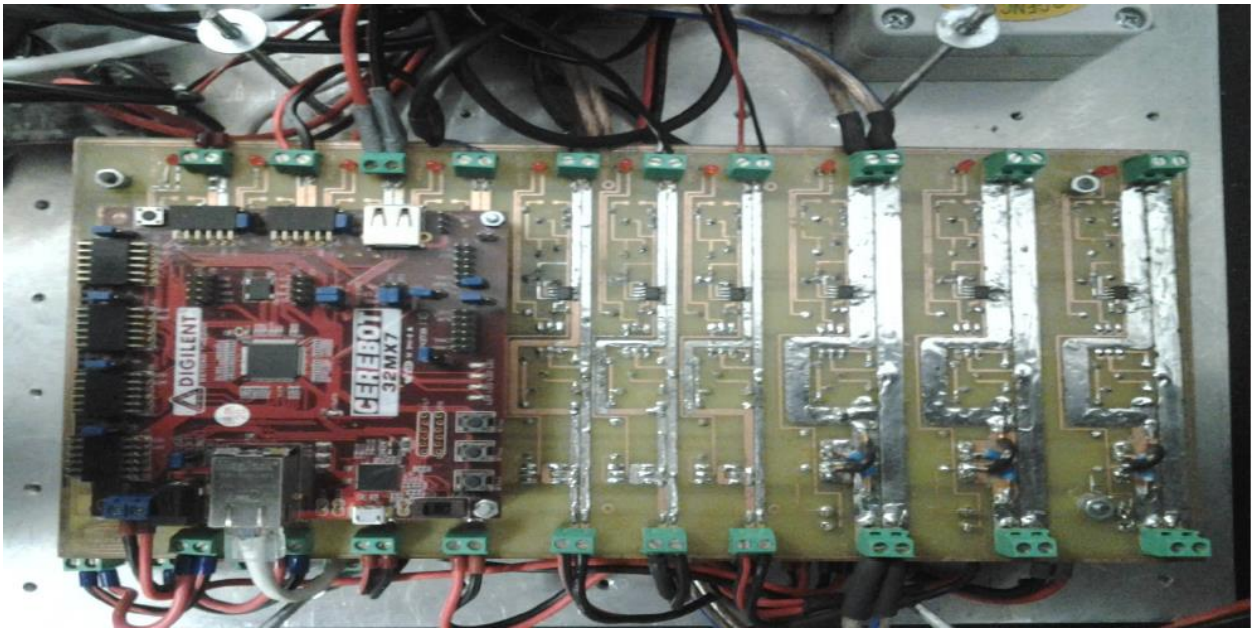
Stikalo (slika 14 spodaj (1)) mora biti vklopljeno, da se izvedejo meritve toka (slika 14 spodaj (2)) in napetosti (slika 14 spodaj (3)), sicer ni porabe na tem kanalu in naprava, povezana s kanalom, je neaktivna. Logika vseh 10 kanalov je enaka, le elektronske komponente so prilagojene na dovoljene moči. V prvi izvedbi diagnostične plošče imamo le sposobnost zaznavanja napetosti in ne tudi izvajanja meritev. Diagnostična plošča je bila zgrajena z eno dvostransko PBC 210 x 297 mm ploščo velikosti A4-formata, ki smo jo izdelali z laserskim tiskalnikom in jedkanjem. Z Digilent Cerebot smo diagnostično vezje povezali, kot je to razvidno na sliki 15.

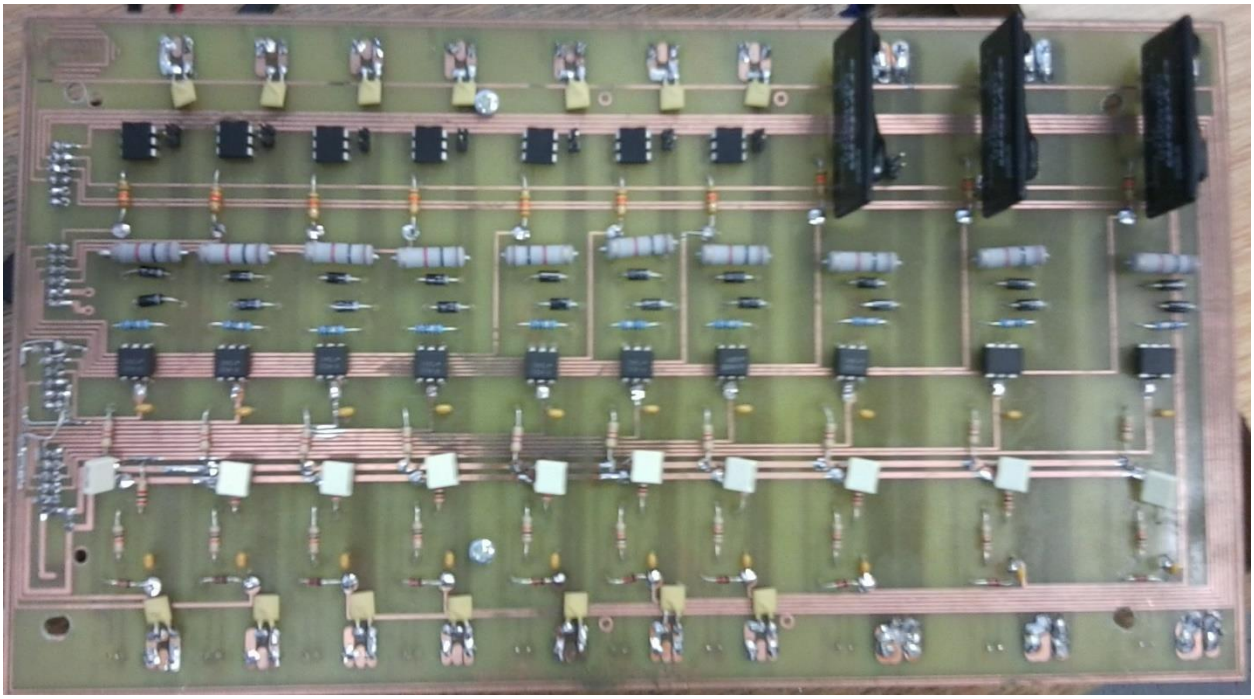
Tabela 1: Seznam elektronskih elementov, uporabljenih v diagnostični plošči

	Naziv:	**Tip:**	**Opis:**	**Količina:**
1.	Digilent Cerebot krmilje	32MX7	glavna procesna enota	1
ON/OFF STIKALO				
2.	Polprevodniški Rele	CMX100D10	močan polprevodniški rele	3
3.	Polprevodniški Rele	PVN013SPBF	šibkejši polprevodniški rele	7
4.	Led svetleča dioda	Katerakoli	signalne diode za indikacijo	10
ZAZNAVA NAPETOSTI				
5.	Senzor toka	4N25	galvansko ločen, detekcija napetosti	10
6.	Upor	1 kOhm 2W	prilaganje napetosti za 4N25	10
7.	Upor	10 Ohm	prilaganje napetosti za 4N25	10

8.	Upor	130 Ohm	prilaganje napetosti za 4N25	10
9.	Dioda	1N4007	prilaganje napetosti za 4N25	20
10.	Led svetleča dioda	katerakoli	signalne diode za indikacijo	10
MERITEV TOKA				
11.	Hall senzor	ACS712	za meritev toka	10
12.	Kondenzator	1 nF	prilagajanje senzorja ACS712	10
13.	Kondenzator	0.1 uF	prilagajanje senzorja ACS712	10
14.	Kondenzator	100 nF	prilagajanje senzorja ACS712	10
15.	Upor	10 kOhm	prilagajanje senzorja ACS712	10
16.	Dioda	1NN444BW	prilagajanje senzorja ACS712	10

Tabela 1 prikazuje seznam vseh elementov, ki smo jih uporabili za izgradnjo diagnostične plošče, elementi so standardni. Končni produkt – izdelana diagnostična plošča z omenjenimi elementi je vidna na sliki 15.



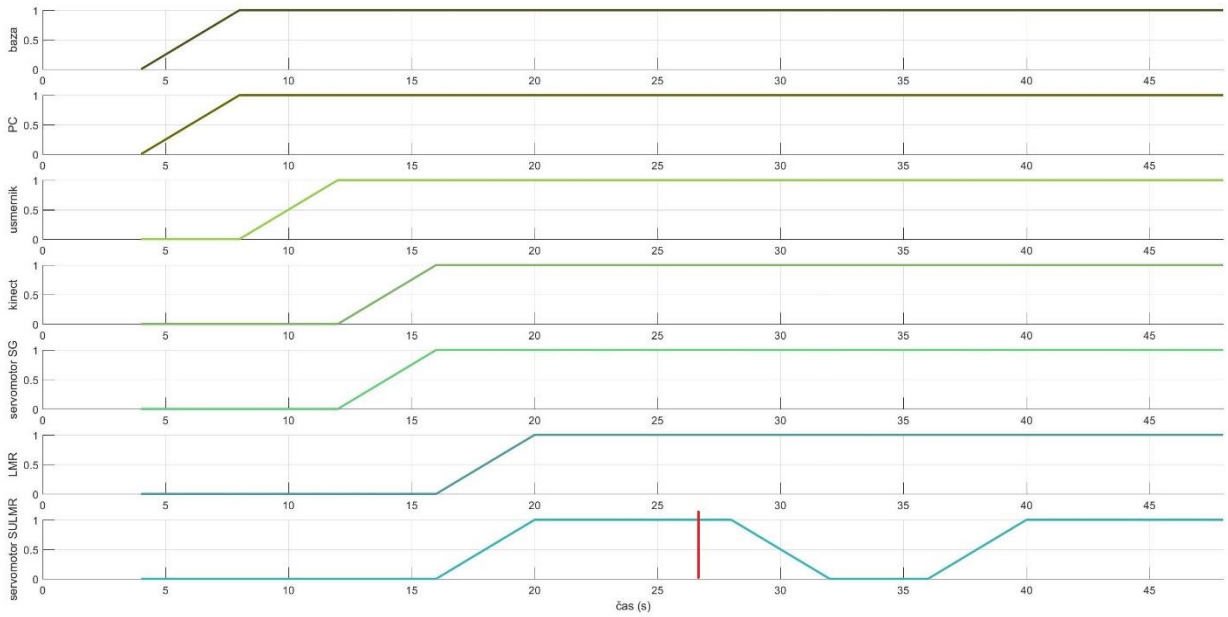


Slika 15: Izdelana diagnostična plošča, zgoraj pogled od zgoraj in spodaj pogled od spodaj.

Diagnostična plošča obsega elektronsko vezje, preko katerega so vezani vsi napajalni sistemi mobilnega robota. To pomeni, da ima diagnostična plošča z vgrajenim mikrokrmilnikom nadzor nad vsemi sistemi. Vsak elektronski sistem je mogoče v tem primeru vklopiti ali izklopiti po potrebi. Ideja se razvija naprej v tako imenovane opazovalce sistema (OS). Opazovalec sistema preverja delovanje sistema, ali ta deluje pravilno. V primeru, da OS opazi napačno delovanje, lahko ponovno zažene določen sistem, s katerim je povezan. Diagnostična plošča ima širok spekter in lahko deluje na različnih sistemih v sodelovanju z opazovalci. Opazovalci lahko posredujejo v delovanje sistemov, z namenom, da izvedejo samo popravljive scenarije. S podatki o napetosti in toku lahko programski del diagnostičnega sistema preverja delovanje sistema, ki je priključen na vejo. Prav tako lahko nadzorni sistem vklopi/izklopi ali ponovno zažene napajalne linije sistema. Pri reševanju naloge smo uspešno implementirali scenarije [bibliografija 7, 9, 10] za detekcijo napak v sistemu in tudi prožili ponovni zagon v primeru zatika servomotorja, ko je ob trku prišlo do preobremenitve in je bila to edina rešitev. Na tak način se lahko brez posredovanja človeka reši morebitno napako na mobilnem robotskem sistemu.

Simulirali smo primer kjer se servomotor na sistemu za uravnavanje LMR zatakne in s tem izgubi funkcionalnost. Slika 16 prikazuje zagon sistemov avtonomnega mobilnega robota. Kjer vrednost 0 predstavlja izklop in vrednost 1 predstavlja vklop napetosti napajanja na določenem sistemu. Vidimo lahko kako se v odvisnosti od časa vklopijo posamezni sistemi. V 27 sekundi smo simulirali zatik

servomotorja na sistemu za uravnavanje LMR (na sliki 16 označen s rdečo črto). Diagnostični sistem je napako prepoznal in jo saniral s krajšim izklopom napajanja na kanalu, kjer je priklopljen servomotor za uravnavanje LMR. Servomotor je ob ponovni pridobitvi napetosti pričel s pravilnim delovanjem.



Slika 16: Časovni potek vklopa sistemov in simulacija zatika servomotorja sistema za uravnavanje LMR.

Hipoteza **H3** je predpostavljala, da bo razvita diagnostična plošča spremljala različne parametre delovanja posameznih sistemov. Predstavljen sistem omogoča spremljanje porabe toka in napetosti na posameznih sistemih, v programskem delu pa je omogoča spremljanje podatkovnih tokov senzorskih podatkov. Na fizičnem nivoju diagnostična plošča omogoča vklop/izklop posamezne veje ali sistema, s čimer je omogočena ponastavitev sistema. Glede na opravljene raziskave lahko potrdimo nekaj scenarijev, kjer smo predstavili diagnosticiranje napake in odpravo te napake. Do sedaj smo na elektronskem sklopu testirali primer, kjer se zatakne servomotor. Zataknitev servomotorja prepoznamo s povečanjem porabe toka, kar diagnostičnemu sistemu javi napako. Prav tako se sproži alarm na programskem delu, kjer diagnostični sistem zazna napako pri podatkovnem toku. Nato sistem sklepa, da je prišlo do fizične preobremenitve sistema za uravnavanje LMR. Diagnostični sistem nato ukrepa s samo popravljivim scenarijem, da za kratek čas izklopi napajalno linijo (glej sliko 16) do sistema za uravnavanje LMR. S tem se ponastavijo parametri servomotorja in le-ta prične spet pravilno delovati. S tem smo dokazali koncept diagnostične plošče in posledično potrdili hipotezo **H3.**

3 PREGLED PODROČJA

Področje avtonomnih mobilnih robotov je zelo obsežno, v našem primeru se osredotočamo na avtonomijo kopenskih in letečih mobilnih robotov. V tem poglavju se posvečamo pregledu področja ter osnovam, ki so pomembne za naše nadaljnje delo.

(a) [3] (b) [30] (c) [76] (d) [77] (e) [78]

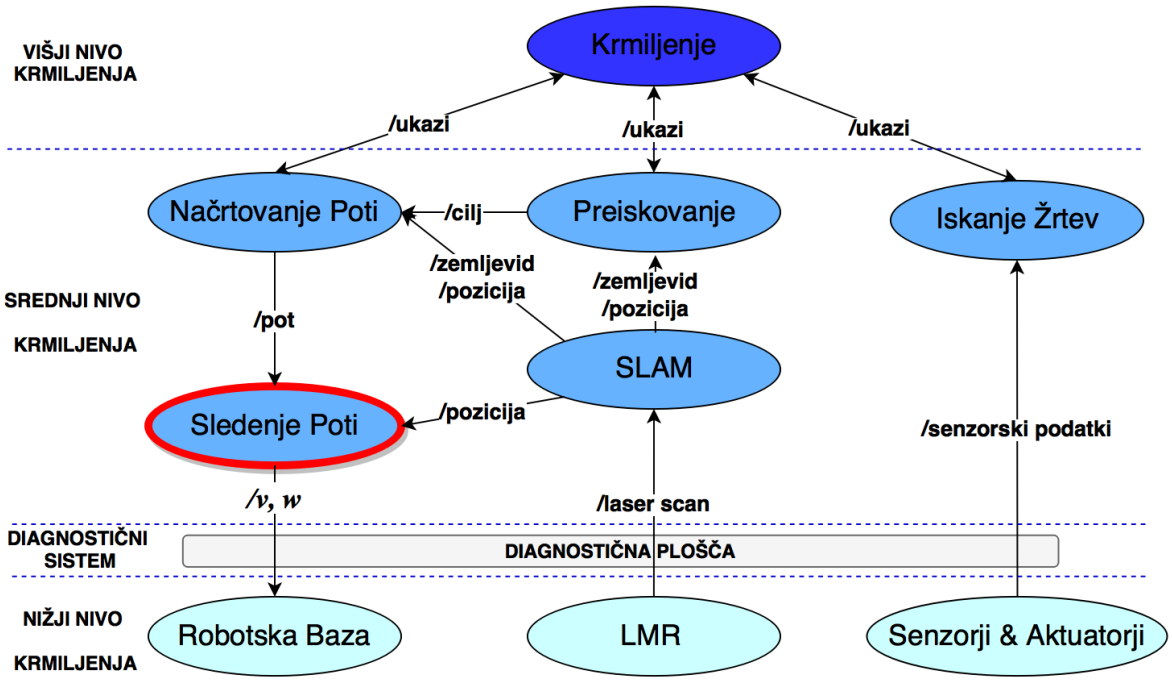
(f) [79] (g) [80] (h) [81] (i) [82] (j) [83]

(k) [84] (l) [85] (m) [86]

Slika 17: Seznam knjig, uporabljenih za raziskovalno delo.

Opravljeno raziskovalno delo na temo avtonomije mobilnih robotov temelji na baznih raziskavah, opravljenih v delih na sliki 17. Literatura na sliki 17 (a–c) vsebuje modele in teoretične razlage osnovnih algoritmov za avtonomno delovanje mobilnih sistemov, vključno s praktičnimi in simulacijskimi primeri. Programski del je bil narejen v robotskem operacijskem sistemu. Osnove orodja za programiranje ROS so

predstavljene v delih (d) in (e)na sliki 17. Teorijo za obdelava slikovnih tokov smo povzeli iz OpenCV, knjigi pod točkama (f) in (g) na sliki 17, kjer so predstavljeni simulacijski primeri, izdelani v Matlab programskem okolju. Dela pod točkami (h), (i) in (j) na sliki 17 smo uporabili za teoretično raziskavo gibanja mobilnih robotov, ta vsebujejo modele, dinamične in kinematične izpeljave. V veliko pomoč pri raziskavi mobilnih robotov na diferencialni pogon so bila dela pod točkama (k) in (l) na sliki 17, saj vsebujejo podatke o gibanju avtonomnih mobilnih robotov v neurejenih prostorih z upoštevanjem trenja in zdrsa. Avtonomni sistemi za nadzor mobilnih robotov in strukturiranih okoljih so zapleteni in modularni. Tukaj se bomo v večjem delu osredotočili na pod-segment nadzora hitrosti avtonomnega sistema, ki je znan kot: izvedba poti, nadzor potovanja, nadzor gibanja in sledenje poti. Celotni sistem avtonomnega navigacije je mogoče videti na sliki 18, povzet iz [3].



Slika 18: Pregled strukture krmilnega sistema avtonomnih mobilnih robotov.

Avtonomni mobilni sistem delimo na več nivojev. Najnižji nivo navigacijskega sistema predstavljajo mehansko-električne komponente, kot so aktuatorji, senzorji in sistem za premikanje mobilnega robota v prostoru. Sledi diagnostični sistem, ki ga na fizičnem nivoju predstavlja diagnostična plošča, namenjena nadzoru in analizi podatkov med delovanjem posameznih sistemov. Srednji nivo avtonomnega sistema predstavljajo algoritmi, ki v celoti tvorijo avtonomni sistem. To so različni algoritmi za akcije, kot je na primer iskanje žrtev. Sistem za kartiranje (SLAM), sistem za iskanje novih raziskovalnih točk v praznem prostoru, sistem za načrtovanje poti do točke preiskovanja in algoritem za sledenje poti. Algoritem za

sledenje poti je močno povezan z robotsko bazo. Sledenje poti je v večji meri odvisno od tipa gibanja mobilne baze, v našem primeru smo se osredotočili na sisteme z diferencialnim pogonom. V nadaljevanju so podrobneje opisani posamezni sistemi avtonomnega navigacijskega sistema mobilnih robotov.

3.1 KRMILNI SISTEMI MOBILNIH ROBOTOV

Mobilni roboti za avtonomno delovanje potrebujejo zaključeno celoto, ki jo imenujemo navigacijski sistem in je grafično prikazana na sliki 18. Navigacijski sistem vsebuje vse potrebne komponente za avtonomno delovanje. V nadaljevanju so podrobneje opisani vsi sistemi in algoritmi, ki so v robotski skupnosti že dobro poznani in že imajo rešitve. Vedno pa ni tako, saj smo v primeru sledenja poti naleteli na težave. Najprej je predstavljen Robotski Operacijski Sistem (ROS), poglavje se nadaljuje z opisi navigacijskih algoritmov.

3.1.1 Robotski operacijski sistem

ROS je dobro poznan v svetu mobilne robotike, je odprtokodni in vsebuje veliko knjižnic za avtonomno mobilno robotiko. ROS smo izbrali z namenom, da se izognemo razvijanju že obstoječih algoritmov, in da nadaljujemo delo drugih in tako prispevamo k avtonomni mobilni robotiki.

Vse omenjene algoritme smo opisali v programskem jeziku C++. Okolje za programiranje sta nam predstavljala Linux Ubuntu in ROS. Program ROS [19], slika 17, točki (d) in (e), je odprtokodni, metaoperacijski sistem za robote. Skrbi za storitve in delovanje strojne opreme, podobno kot ostali operacijski sistemi. Delovanje temelji na izmenjavi sporočil med procesi:

Koncept ROS-a delimo na:

1. **PODATKOVNI NIVO:** ROS podatkovne strukture na disku.
2. **PORAZDELJEN SISTEM PROCESIRANJA**: omrežje procesov ROS.
3. **JAVNI NIVO:** dostopen vsem, vsepovsod.

Struktura ROS:

- **PAKETI:** paketi so glavne organizacijske enote programov (knjižnice, podatkovni seti, nastavitvene datoteke)
- **ODVISNOSTI:** manifesti (priskrbijo meta podatke o paketih; odvisnosti, prevajalnik, zastavice ...)
- **PAKETI:** podatkovne strukture so zbirke paketov, ki v celoti tvorijo neko funkcionalnost, npr.: navigacijski paket

- **PAKETNE ODVISNOSTI:** podatkovni manifesti priskrbijo podatke o podatkovnih strukturah, npr.: odvisnosti od drugih podatkovnih struktur
- **SPOROČILA:** opisi sporočil, definirajo podatkovno strukturo za sporočila, ki jih bo ROS poslal
- **STORITVE:** opisi storitev, definirajo podatkovno strukturo za prošnje in odgovore storitev v ROS-u

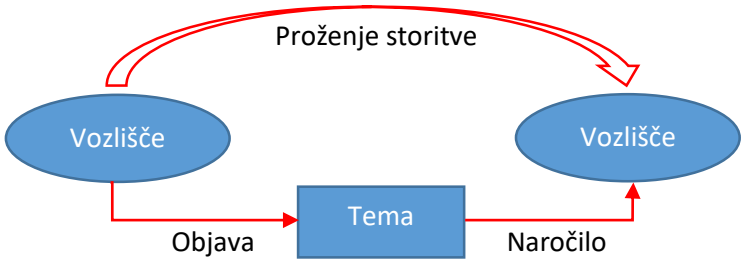
Vozlišča so procesi, ki izvajajo procesiranje. ROS je modularen, saj robotski kontrolni sistem navadno sestavlja več procesov. ROS vozlišče je zapisano v C++ ali Python programskem jeziku.

Upravljalec skrbi za registracijo posameznih vozlišč in ima pregled nad ostalimi procesi. Brez upravljavca se vozlišča ne bi znašla med seboj. Upravljavec beleži informacije posameznih ROS vozlišč.

Strežnik za parametre dovoljuje shranjevanje podatkov v centralni lokaciji na podlagi ključev. Parametri so mrežno dostopni, globalni. So del upravljavca in z njimi nadziramo procese v času delovanja.

Sporočila: vozlišča komunicirajo med seboj z izmenjavo sporočil.

- Založnik/naročnik: veliko število, različna vozlišča, različni tipi.
- Sporočilo je podatkovna struktura s tipom:
    - črka ali majhno celo število,
    - celo število (angl. *integer*),
    - plavajoča vejica,
    - booleovska vrednost,
    - dvakrat natančnejša plavajoča vejica,
    - več črk v nizu.
- Sporočila lahko vključujejo arbitražno razporejene strukture in vektorje.



Slika 19: Grafični prikaz sporočil in storitev med vozlišči.

Vozlišče pošlje sporočilo z določeno Temo.

- Tema je ime, s katerim se identificira vsebina sporočila.

- Vozlišče, ki ga zanima vsebina sporočila, se naroči na določeno Temo.
- Založniki (angl. *publisher*) in naročniki (angl. *subscriber*) se ne zavedajo eden drugega.
- Lokalno, lahko se objavi Tema in vsakdo se lahko nanjo naroči, prav tako lahko vsako vozlišče objavi novo sporočilo, vendar morajo biti komunikacijska sporočila istega tipa.
- Storitve: založnik (angl. *publisher*)/naročnik (angl. *subscriber*): veliko število, različna vozlišča, različni tipi.
- Zahteva/odgovor: storitve.
- Pari sestavljenih sporočil: eno za zahtevo in eno za odgovor.
- Vozlišče, ki je založnik storitev, skrbi za storitve pod določenim imenom in naročnik uporablja to ime za pošiljanje zahtev ter nato čaka na odgovor.

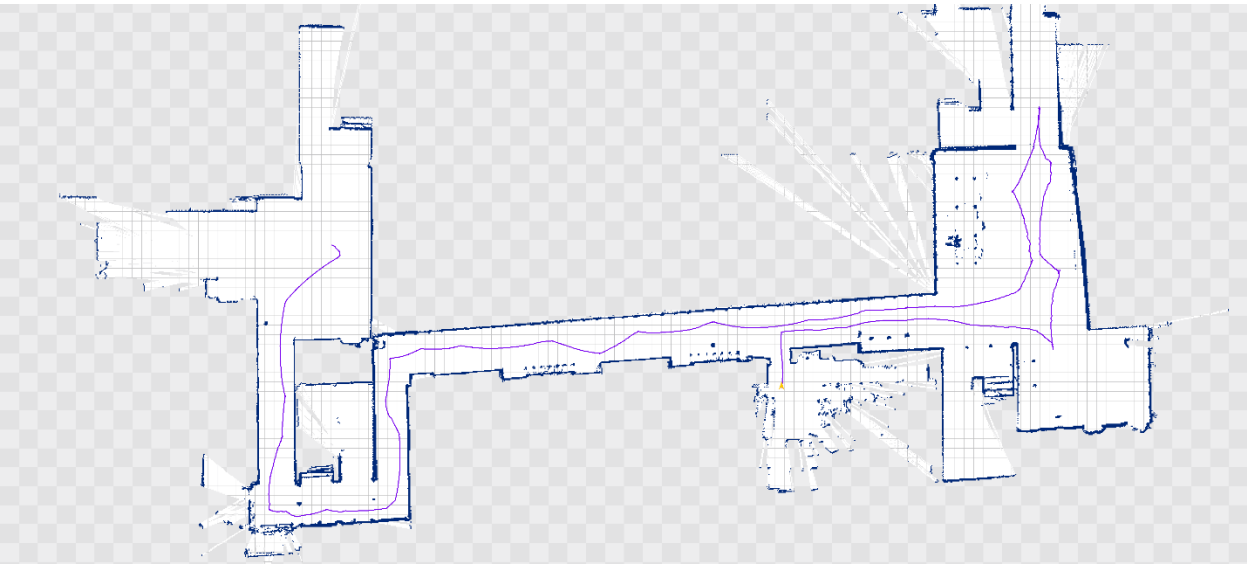
ROS postaja vedno širše uporabljen operacijski sistem za robote, prav tako se širi na druga področja, kot so krmiljenje robotov v industriji, manipulacija in področje avtonomnih letečih robotov.

3.1.2 Kartiranje in lokalizacija

Osnova za vse navigacijske algoritme je model okolja, tega z 2D-senzorjem izdelamo s SLAM postopki [22–24]. V 3D-prostoru je potrebno podrobno izdelati model okolja, ta tema je obravnavana v poglavju 4. V našem delu smo se najprej srečali s tako imenovanim SLAM (angl. *simultaneous localization and mapping*) problemom. Problem hkratne lokalizacije in gradnje zemljevida predstavlja osnovo za naslednje navigacijske sisteme, kar pomeni, da je zgrajen zemljevid zelo pomemben in mora biti dobro pripravljen. Pri našem delu smo iskali postopek SLAM, ki deluje brez odometrije in brez zunanjih lokalizacijskih sistemov. To pomeni, da se mora robot zanašati na svoje senzorje. Z LMR senzorjem lahko zgradimo zemljevid in se lokaliziramo s postopki SLAM, kot so Gmapping [20, 21], FastSLAM, Hector Mapping in podobno. Tukaj bomo predstavili izbran sistem Hector SLAM [22, 23], ki se je izkazal kot najbolj učinkovit sistem za kartiranje in lokalizacijo.

Algoritmi, ki jih poimenujemo s SLAM, sestavljajo po navadi dve komponenti. Prva je tako imenovana samodejna lokalizacija, ki je hkrati povezana s kartiranjem. Pri algoritmih, ki jih poimenujemo kartiranje (angl. *mapping*), imamo v mislih samo kartiranje, brez lokalizacije. V tem primeru je treba priskrbeti drug vir podatkov o lokaciji, saj je od tega odvisna kakovost gradnje zemljevida. V primeru SLAM algoritem s seštevanjem majhnih premikov ocenjuje absolutno pozicijo robota v prostoru. Algoritem je v tem primeru v celoti odvisen od LMR, kar je v primerih, ko ni mogoče uporabiti drugih senzorjev za oceno pozicije, zelo

uporabno. Ker smo v našem delu v večini primerov uporabljali Hector Mapping, ga bomo v nadaljevanju podrobneje predstavili.



Slika 20: Primer zemljevida okolice, izdelan na FERI (merska enota kocka 1 x 1m).

Algoritem za kartiranje Hector SLAM ali Hector Mapping deluje samo na podlagi podatkov iz laserskega merilnika razdalje, glavni način lokaliziranja robota v prostoru je z metodo ujemanja z optičnim branjem (angl. *scan-matching*). S postopkom se poiščejo ujemajoči se deli optično prebranega območja. Opisani algoritem je bil razvit za kartiranje »RoboCup NIST aren« in je bil razvit s strani nemške ekipe Darmstadt. V nadaljevanju sledi podrobnejši opis omenjenega algoritma.

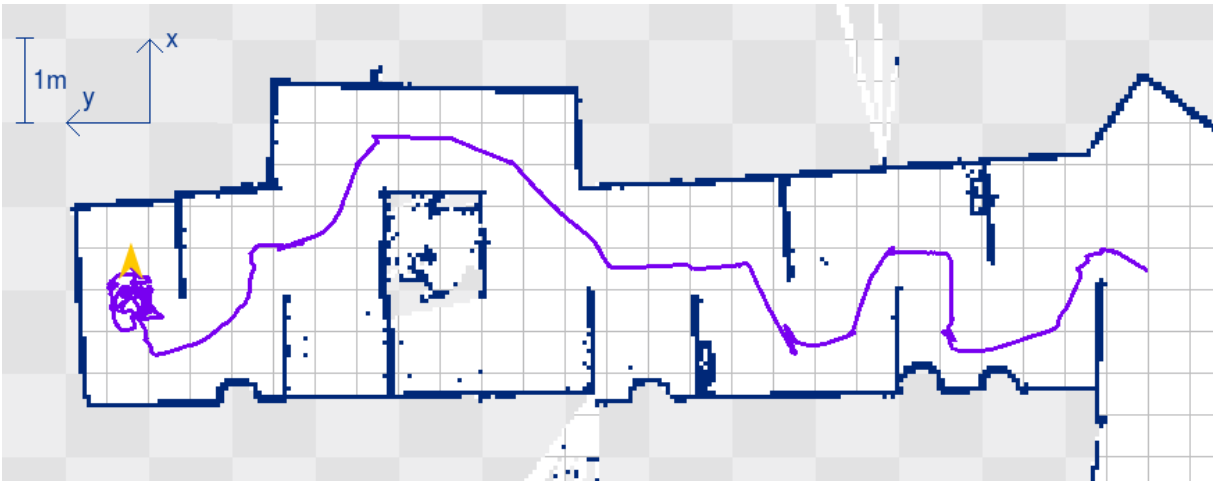
Vhodni podatki: meritve laserskega merilnika razdalje: /scan**,** transformacije glede na postavitev senzorja: /tf**,** transformacije na robotu

Izhodni podatki: zemljevid: /map**,** pozicija robota na zemljevidu

Koordinatni sistemi: odometrija ni potrebna, potrebujemo transformacije od izhodišča robota do laserskega merilnika razdalje

Zemljevid je predstavljen v 2D-mreži z različnimi barvami, te predstavljajo zasedenost posamezne celice. Velikost celice zemljevida je definirana z resolucijo, ki jo poda uporabnik. Barva celic je definirana kot 8-bitna vrednost, –1 (siva) predstavlja neznan prostor; 0 (črna) predstavlja ovire na poti; 125 (svetlo siva) predstavlja že raziskan prostor. Možen je hiter dostop do zemljevida.

Hector mapping nepretrgoma izvaja naslednje faze pri izdelavi zemljevida. Najprej se zajamejo podatki iz laserskega merilnika razdalje, nato se podatki laserskega merilnika transformirajo v izhodiščni koordinatni sistem mobilnega robota, zatem se izvede filtriranje, s katerim se omeji vidno polje ter maksimalne in minimalne razdalje, ki so lahko uporabljene za kartiranje. Sledi ocena 2D-pozicije v prostoru. Ocena se izvede z začetnim izhodiščem ali pa glede na prejšnjo obstoječo ali ocenjeno pozicijo. Točke se najprej projicirajo na obstoječ zemljevid glede na trenutni položaj, sledi ocena verjetnosti zasedenosti končnih točk. Nato se izvede odvod slike za ugotovitev položaja po metodi Gauss-Newton. To se izvede v več nivojih z različno resolucijo zemljevida, predvsem zaradi odprave napak in nepravilnosti položaja, saj pri manjši natančnosti lahko v določenih primerih odvod vrne boljše rezultate. Zaključi se z osvežitvijo zemljevida, pod pogojem da je robot prepotoval določeno razdaljo. Shranjevanje podatkov preteklih pozicij robota, ki so objavljene na temo je uporabno za predstavitev poti robota. GeoTiff je standardna oblika zemljevida za shranjevanje, poslati moramo ukaz na določeno temo v sistemu.



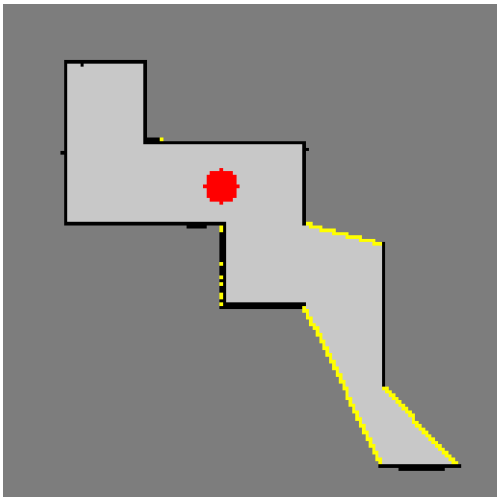
Slika 21: Oblika formata, v katerem se zemljevid shrani.

Algoritem Hector Mapping lahko upošteva tudi višinsko kartiranje, ki se izvaja na podlagi 3D-senzorja, kot je na primer senzor Kinect. Višinsko kartiranje na podlagi geometričnih parametrov izvede transformacijo zaznane ovire na 2D-zemljevid, ki se združi z opisanim pristopom. S tem dobimo dodatno informacijo o okolici, prikazano kar v 2D-zemljevidu. Sistem je zelo uporaben v primeru miz in polic, ko mobilni robot ne more izpeljati vožnje. Primer razvitega algoritma za kartiranje kompleksnih okolij je mogoče videti v [24].

3.1.3 Preiskovanje

Preiskovanje okolice je naslednji logični korak v navigacijskem sistemu. Sedaj je treba na podlagi izdelanega zemljevida določiti naslednjo raziskovalno točko v prostoru. Preiskovalni algoritem določi nov

cilj v prostoru, ki naj bi ga robot dosegel. Če to ni mogoče, je treba določiti nov preiskovalni cilj. Zanimiv pristop preiskovalne tehnike je predstavljen v [2], avtorji na podlagi ocene zanesljivosti senzorjev ocenijo tveganje za dosego preiskovalnega cilja. V nadaljevanju bomo opisali enostaven in najpogostejši pristop preiskovanja. Preiskovanje mobilnih robotov je omejeno na znani zemljevid. Običajno se uporabljajo tehnike temelječe na raziskovanju novih področjih. (angl. *frontier-based*) [25]. Ti algoritmi imajo za osnovo zemljevid, na katerem se iščejo meje med znanim in neznanim prostorom. Primer raziskovalnega postopka je prikazan na sliki 20. Meje, ki so zanimive za preiskovanje, so obarvane rumeno.



Slika 22: Tehnika preiskovanja prostora, slika iz vira [26].

Preiskovanje je omejeno na znani zemljevid. Zemljevid se med prikazovanjem robota spreminja, to pomeni, da se premikajo tudi raziskovalne meje, novi cilji se pojavljajo vse dokler ni več možnosti raziskovanja. Raziskovalni algoritem nam vrne ciljno točko v danem trenutku, imenujemo jo tudi preiskovalna točka. Na ta način lahko celovito zaključimo preiskovanje prostora. Podrobnejši opis pristopa je mogoče prebrati v diplomskem delu [26].

3.1.4 Načrtovanje poti

Načrtovanje poti je definirano kot povezovanje preiskovalne točke s trenutnim položajem robota. V danem zemljevidu želimo najti prosto pot od trenutnega položaja robota do raziskovalne točke. Načrtana pot se predstavi kot serija premic, ki v prostoru določajo točke, katerim naj bi robot sledil za uspešen doseg preiskovalnega cilja. Za izračun poti obstaja veliko znanih algoritmov, ki so obravnavani v poglavju 3.2.

3.1.5 Sledenje poti

Do te točke smo zgradili zemljevid okolice, določeno imamo raziskovalno točko in pripravljen načrt, kako doseči to točko z dano mobilno robotsko bazo. Zadnji korak je sledenje načrtani poti. Sledenje je tesno povezano z dinamiko mobilnega robota. V našem primeru smo se osredotočili na sledenje poti z mobilnim robotom na diferencialni pogon, to je predstavljajo zadnji korak v zanki navigacijskega algoritma in se izvaja se vedno, ko je na voljo načrtana pot v prostoru. Sledenje poti ima velik pomen v navigacijskih algoritmih, vendar se predstavljeni sistemi navigacije ne spuščajo v izdelavo teh algoritmov. Izdelava algoritma za sledenje je kompleksno, saj je treba upoštevati dinamiko različnih robotov, kakor tudi integracijo algoritma za sledenje v obstoječ sistem. Vse vodi do uspešnega sledenja, ki mora biti hitro in enostavno za uporabo. Del izvedbe poti je naše glavno raziskovalno delo, pristop, ki smo ga uporabili na naših robotih, je opisan v poglavju 4.

3.2 SORODNA DELA NA TEMO SLEDENJA POTI

Naše delo na področju avtonomne mobilne robotike se je pričelo z obstoječimi odprtokodnimi sistemskimi rešitvami, kot je navigacijski paket Hector, ki ga je razvila ekipa iz Darmstadta v Nemčiji. Programska oprema Hector vsebuje sočasno lokalizacijo in kartiranje (SLAM) [22–24]. Poleg tega Hector zagotavlja paket za mobilne robote Hector_navigation [27–29], ki vključuje cenitveni zemljevid in višinsko kartiranje, raziskovanje in načrtovanje poti. Kartiranje in raziskovanje Hector se je večkrat izkazalo na tekmovanjih RoboCup, kot primer izpostavimo »RoboCup 2013«, kjer se je izkazal kot najboljši v razredu »Autonomy«. Programska oprema deluje zelo dobro, zato smo ga izbrali za našega robota. Za zaključek avtonomnega sistema smo potrebovali tudi sledenje poti. Programski paket je sicer tudi na voljo v navigacijskem paketu Hector, vendar smo po nekaj testnih vožnjah opazili, da so se pojavile težave pri sledenju. Mobilni robot je v tem primeru nepravilno sledil začrtani poti in trčil v oviro. Na podlagi tega smo ugotovili, da sledenje poti Hector ni povsem primerno za gosenične mobilne robote. Zaradi teh dejstev smo se odločili, da razvijemo lastni algoritem za sledenje poti, ki je bil nato tudi preizkušen na tekmovanjih RoboCup.

Splošni algoritmi za sledenje poti in nadzor gibanja so opisani v [30] in obravnavajo vozila, podobna avtomobilom. Opisani pristopi se lahko uporabijo za mobilne robote na gosenični in diferencialni pogon, ki so zmožni obračanja na mestu. Delo, predstavljeno v [31] in [32], govori o načrtovanju gladkih poti za mobilne robote na diferencialni pogon. Predstavljene so kontrolne metode za usmerjanje mobilnih robotov, vključno s preverjanjem trkov. Delo v [33] se osredotoča na gladko gradnjo poti s predhodno definiranimi opornimi točkami v prostoru. V tem poizkusu je bil uporabljen kinematični model mobilnega

robota. Avtorji so poskušali optimizirati razdaljo ali čas potovanja, da bi dobili najboljšo možno pot za mobilnega robota z upoštevanjem minimalnega radija obračanja. Primerjave med različnimi tipi algoritmov so predstavljene kot rezultati simulacij. V [34] avtorji niso opravili praktičnih preizkusov, temveč so med simulacijo poskušali popraviti položaj robota glede referenčno podane krožnice. Korekcija je bila izvedena z novo generirano, pravilno usmerjeno krožnico, ki se seka z referenčno krožnico. Delo, opravljeno v [35], je bilo izvedeno v dveh fazah; v prvem delu je bila izvedena optimalna povezava trenutnega položaja robota in referenčne trajektorije. V drugem koraku pa tako imenovana premična odvisnost, ki vodi robota na želeno pot. Realne težave z izvedbo poti so opisane v [36], kjer pojasnjujejo, zakaj bi bilo njihovo delo primerno za robote z visokimi hitrostmi. Avtorji trdijo, da je idealno sledenje poti odvisno od natančnosti ocene trenutne pozicije mobilnega robota. V [37] je predstavljen algoritem za generiranje poti s pomočjo krivulj Bezier, kjer se uporabljajo ti vzorci hitrosti, in je izvedba poti optimizirana s parametri. Odprto kodni sledilnik poti Hector_path_follower, ki je, ko že povedano, del navigacijskega paketa Hector, temelji na algoritmu pose_follower [38], ki je del ROS-a. Osnovan je na preprostih izračunih razlik translacije in rotacije. Linearna in kotna hitrosti se izračunata z uporabo preproste transformacije, množenje razlike oddaljenosti od trenutne do naslednje točke poti in parametra vpliva, ki ga nastavi uporabnik. Izračunane hitrosti so omejene z mejnimi vrednostmi in omejenostjo vrtenja na mestu. Preverjajo se tudi možnosti trka na poti. V naših raziskavah smo zasledili sorodno temo v [40, 41], kjer so predstavili tehniko čistega zasledovanja, ki temelji na preprostem proporcionalnem krmilniku z uporabo napake premika in parameter pogleda vnaprej (angl. *lookahead*), ki je definiral naslednjo ciljno točko v prostoru. Hitrosti so izračunane na podlagi geometrijskih izračunov potrebnega polmera za doseg ciljne točke. Avtorji so odkrili, da se njihov algoritem obnaša zelo dobro, predstavljen je bil kot stabilnejši in natančnejši način za sledenje poti. Sledenje poti je pogosto uporabljena tehnika, vendar je zaslediti pomanjkanje napredka pri teh preprostih in robustnih algoritmih, kar bo glavna tema našega dela. Osnovni problem je sledenje dani poti z razpoložljivim mobilnim robotom. Naš cilj pa je ustvariti algoritem, ki se lahko uporablja na robotih z diferencialnim pogonom in je enostaven za uporabo.

Naslednje podobno delo se imenuje Dynamic Window Approach (DWA). Omenjeni pristop ima prednosti pred klasičnim sledenjem poti, ker vključuje preverjanje trka na poti. Preverjanje trkov je potrebno v dinamičnih okoljih, kar pa povečuje kompleksnost algoritma. Programska oprema DWA je na voljo v ROS pod imenom move_base. Njegove izvedbe so opisane v [42–50] in lahko sloni na principu DWA ali Trajectory Rollout (TR). Oba pristopa imata kontrolni prostor vzorčenja z razpoložljivimi hitrosti, razlikujeta se glede na način, kako se vzorči hitrostni prostor. TR izvede vzorčenje iz nabora vseh dosegljivih hitrosti

na vsem razpoložljivem okolju, DWA vzorči samo v obsegu dosegljive hitrosti. DWA je primeren za neurejena okolja, vendar potrebuje dodatnega globalnega načrtovalca poti.

DWA neposredno išče potrebne krmilne signale za vožnjo robota v prostoru z možnimi hitrostmi. Okno z možnimi hitrostmi se omejuje na tiste hitrosti, ki so dinamično mogoče ter varne; torej se ne dotaknejo sten. Zmanjševanje mogočih hitrosti je narejeno v prvem delu, v drugem z objektno funkcijo optimizira hitrosti. V tem primeru obravnavamo robote s sinhronim pogonom, to pomeni, da za njegovo vožnjo potrebujemo translacijsko in kotno hitrost, v nadaljevanju jih omenjamo pod skupnim imenom hitrosti. V našem primeru smo algoritem ponovili v programskem okolju Matlab, s katerega so tudi slike iz nadaljevanja.

Prvi del algoritma DWA:

1. Izračun trajektorij

   Vsaka krivulja je unikatno definirana glede na kotno in linearno hitrost, ta pa je sestavljena iz odsekov krožnic, po katerih se giblje robot. Hitrosti morajo biti določene v nekem časovnem intervalu t. Pod pogojem, da trajektorijo na preseka ovira. Pri izračunih trajektorij upoštevamo samo prvi časovni interval, na katerem predpostavimo, da so hitrosti konstantne. Polmer trajektorije določimo z $R = v/\omega$.

2. Izračun dovoljenih hitrostih

   Bližnje ovire omejujejo obračanje in gibanje naravnost. V tem delu upoštevamo samo hitrosti, pri katerih se robot lahko varno ustavi. Dovoljene hitrosti so tiste, katere:

$$V_a = \left\{(v,\omega)|\ v \leq \sqrt{2 * dist(v,\omega) * v_b} \wedge w \leq \sqrt{2 * dist(v,\omega) * \omega_b}\right.$$

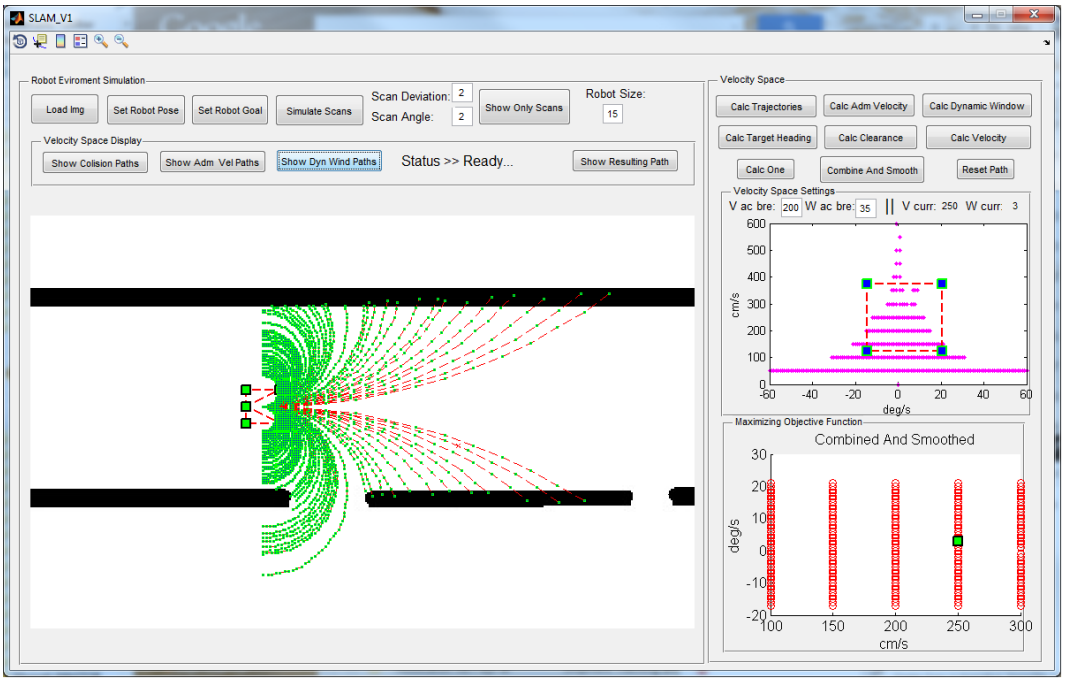
   ,kjer je $dist(v,\omega)$ razdalja do najbližje ovire in sta $v_b\ ter\ \omega_b$ zaviralna pospeška.

3. Izračun dinamičnega okna

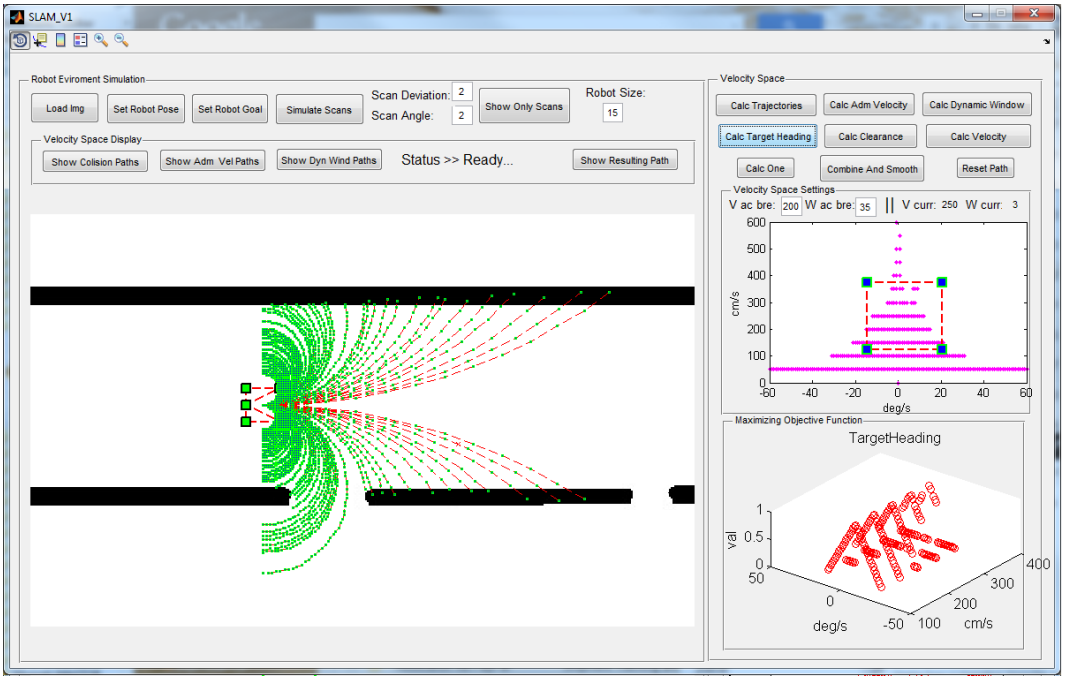
   Z upoštevanjem omejenega pospeška hitrosti in trenutne hitrosti robota se prostor hitrosti zmanjša na tako imenovano dinamično okno, ki vsebuje hitrosti, dosegljive v naslednjem časovnem intervalu. Dinamično okno se nahaja okoli centra trenutnih hitrosti, vse ostale krivulje izven okna niso dosegljive glede na omejitve.

$$V_d = \{(v,\omega)|\, v \in [v_a - \dot{v} * t, v_a + \dot{v} * t] \wedge \omega \in [\omega_a - \dot{\omega} * t, \omega_a + \dot{\omega} * t]\} \quad (28)$$

,kjer sta v_a in ω_a trenutni hitrosti, $\dot{v}$ in $\dot{w}$ maksimalni pospeški, ki določajo velikost dinamičnega okna.



Slika 23: Dinamično okno v prostoru hitrosti.



Slika 24: Levo ovrednotene smeri in desno primer izračuna kota.

Drugi del algoritma DWA:

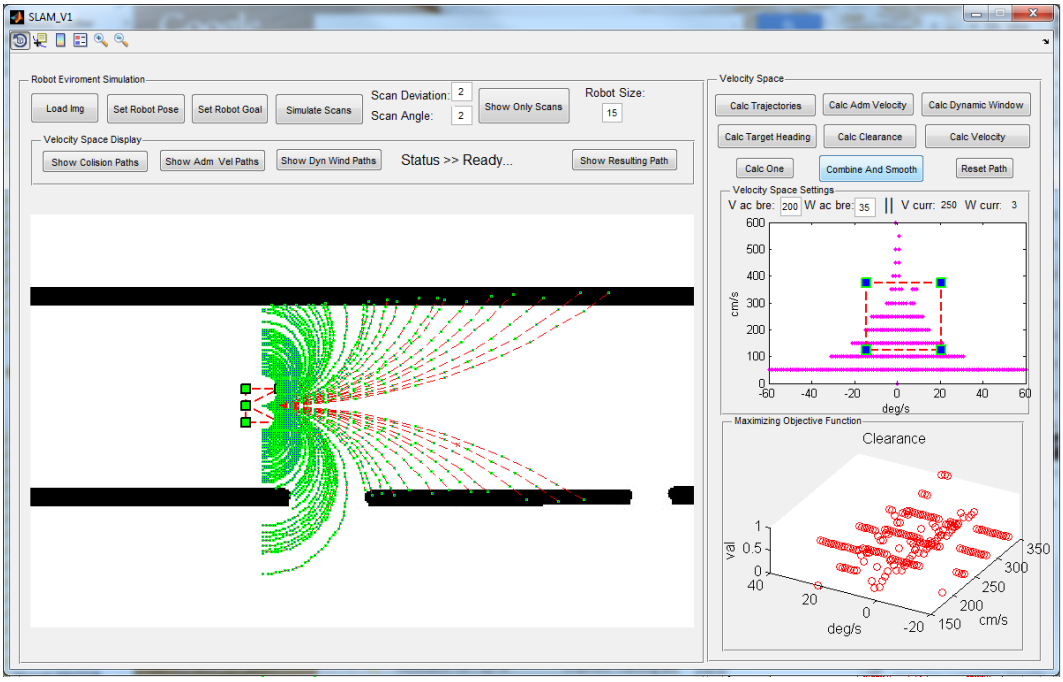
Ocenjevanje vseh treh funkcij se izvede na intervalu [0, 1], kjer 1 predstavlja najboljšo možnost. Gre za polje točk, ki izhajajo iz vrednosti $[v, \omega]$. V našem primeru govorimo o diskretnih vrednostih, zato tudi grafe prikazujemo s točkami. Spodnje slike so prikazane zvezno. Imamo tri polja z vrednostmi v tretji dimenziji, v nadaljevanju jih imenujemo tudi konstante polja hitrosti $[v, \omega]$.

1. Določitev smeri cilja

   Smer cilja nam pove, koliko je robot poravnan s ciljem, dan je z 180-theta, kot je prikazan na sliki 23. Ker se smer spreminja z različnimi hitrostmi, je ta izračunana samo za prvi časovni interval. Predpostavimo, da so hitrosti konstantne.

2. Določitev prehodnosti poti

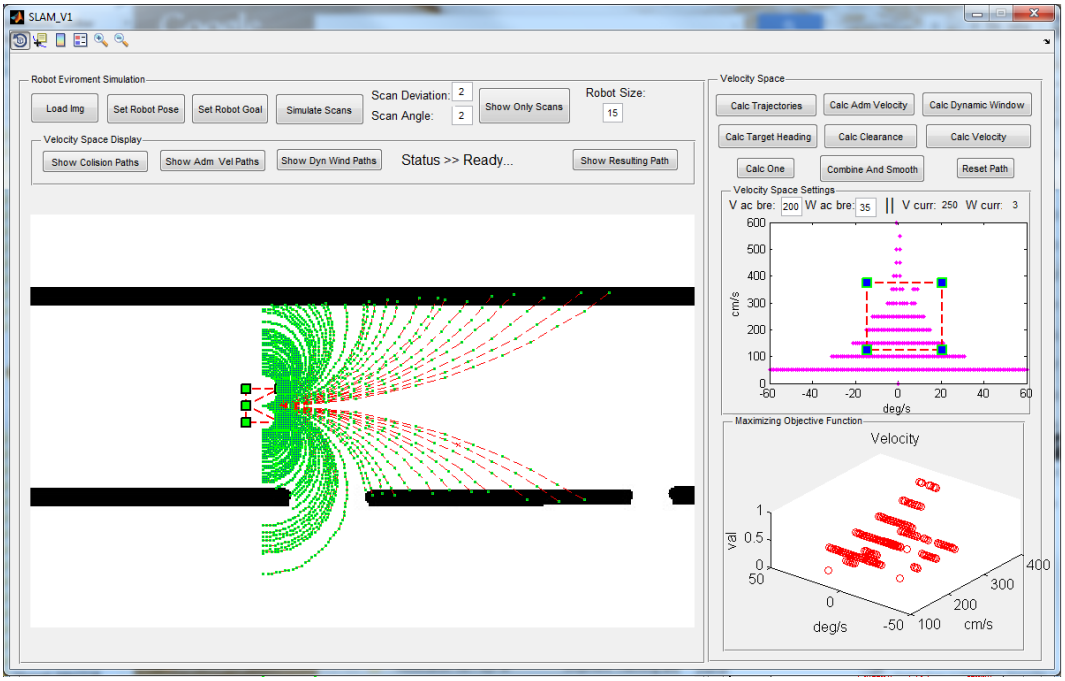
   Če na poti robota ni nobene ovire, je konstanta postavljena na 1, drugače se manjša proti 0. Računa se kot, ki predstavlja razliko do 180°, to prepreči, da bi se izračunal negativni kot.



Slika 25: Levo ovrednotenje prehodnosti poti in desno primer izračuna razdalje.

3. Določitev hitrosti

   Prikaz projekcij dovoljenih hitrosti, kot prikazuje slika 26.



Slika 26: Ovrednotene hitrosti.

<u>Glajenje in določitev rezultirajoče hitrosti</u>:

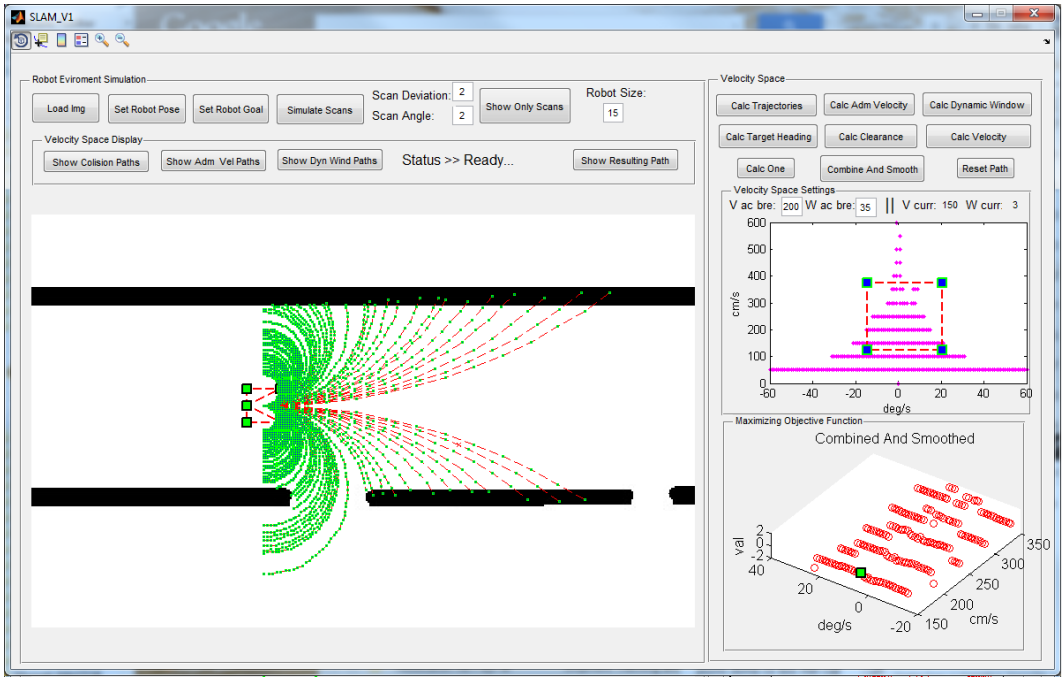
S pomočjo rezultatov iz prvega dela smo ocenili, katera možna pot bi predstavljala najboljšo v treh pogojih z danimi rezultati:

1. smeri: $smer(v, \omega)$
2. prehodnosti: $razdalja(v, \omega)$
3. hitrosti: $hitrosti(v, \omega))$

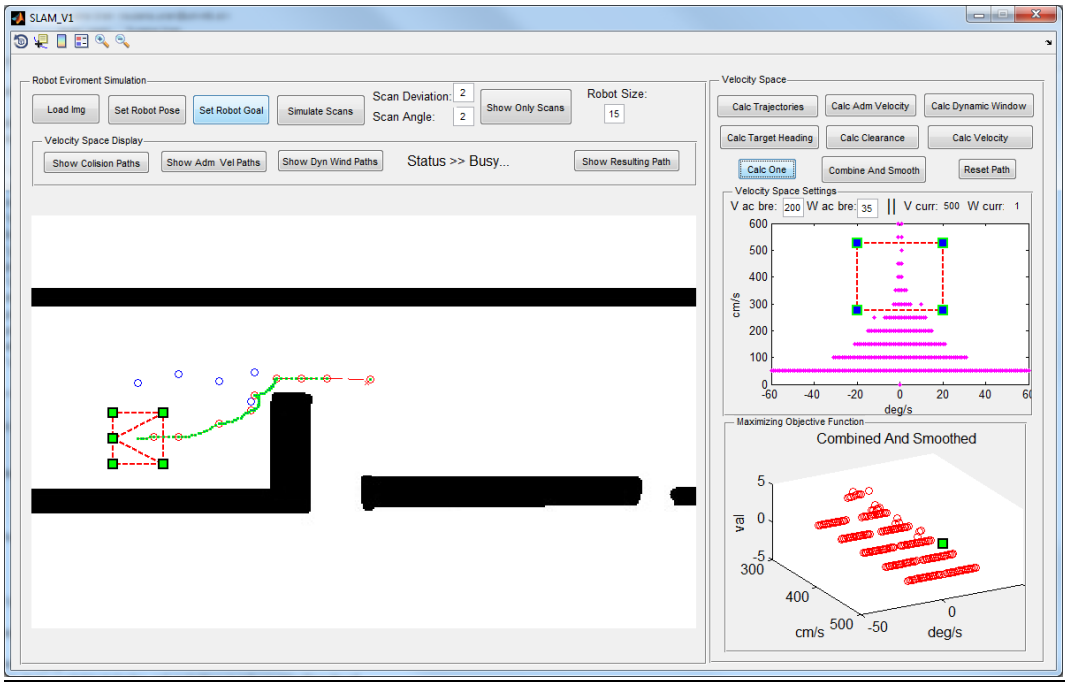
Objektna funkcija za združevanje rezultatov je:

$G(v, w) = \max(\ \sigma(\alpha * smer(v, \omega)\ +\ \beta * razdalja(v, \omega) + \gamma * hitrosti(v, \omega)))$, kjer je $G(v, \omega)$ maksimum rezultirajoče funkcije glede na objektno funkcijo.

Parametri $\sigma,\ \alpha,\ \beta, \gamma$ so lahko določeni eksperimentalno, priporočene vrednosti pa so 1, 0.8, 0.1, 0.1.



Slika 27: Rezultat in določitev hitrosti.



Slika 28: Primer potovanja robota z DWA.

Odvisnosti:

- Velikost dinamičnega okna je v veliki meri odvisna od dovoljenih pospeškov.
- Če se robot zagozdi, se reši tako, da se vrti na mestu, dokler spet ne pelje linearno.
- Hitrost potovanja robota se določi glede na njegovo oddaljenost od ovir v danem trenutku.

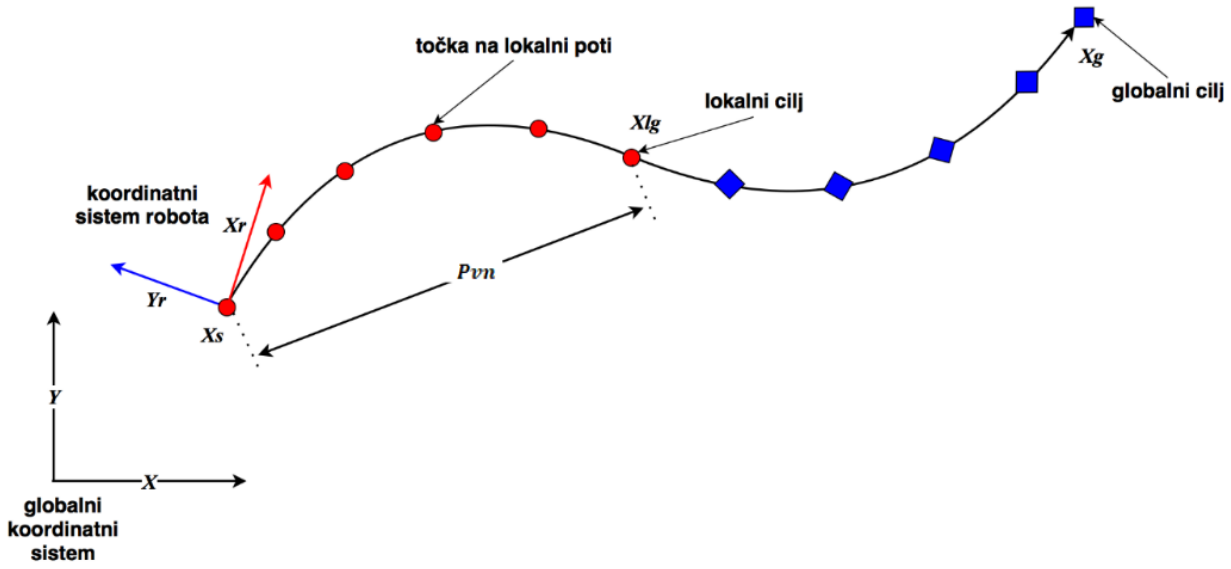
Primer na sliki 28 prikazuje izračun in izvedbo poti v primeru uporabe pristopa DWA. Na sliki lahko vidimo, da je bilo izračunanih 5 centrov krožnic, označenih z modrim krogom. V primeru potovanja v ravnini je radij potovalne krožnice zelo velik ali neskončen in ga ni mogoče prikazati na sliki. V omenjenem primeru v simulaciji se je robot na sliki uspešno izognil oviri in izvedel pot do začrtanega cilja, kar pa je zahtevalo ogromno procesorske moči in časa. Ostale sorodne članke na temo sledenja poti je mogoče videti v [51–54], v katerih avtorji v večini primerov uporabljajo lastne algoritme. Ti omejujejo hitrosti in začasno generirajo trajektorije v obliki krožnic.

4 ALGORITEM ZA SLEDENJE POTI

Osnovni koncept sloni na kartiranju ali izdelavi načrta prostora (poglavje 3.1.2), na podlagi katerega se izvede preiskovanje prostora in določitev globalnih ciljev. Na podlagi slednjih, danega zemljevida ter trenutne pozicije robota se izdela pot do globalnega cilja. Znana pot je poslana algoritmu za sledenje poti, kjer se izvedejo in izračunajo trenutne hitrosti. V tem delu se posvečamo samo sledenju poti, ki je zelo pomembna pri realizaciji avtonomnega robotskega sistema. Z algoritmom za sledenje poti z dinamičnim prilagajanjem krožnih lokov (DPKL) želimo potrditi predpostavke hipoteze **1 (H1)**.

4.1 OPIS ALGORITMA ZA SLEDENJE

Algoritem za sledenje načrtane poti izračunava trenutne krmilne hitrosti za mobilno bazo. Robot z gosenicami se obnaša kot dvokolesni robot, njegove kinematične enačbe so zapisane v poglavju 2.2. Za premikanje robota v različnih smereh potrebujemo linearno in kotno hitrost. Imamo podano načrtano pot v 3D- ali 2D-prostoru z aktualnim zemljevidom. Trenutna lokacija robota je x_s. Za natančno sledenje trajektorij potrebujemo točno lokacijo robota, to dobimo s pomočjo transformacij, ki jih priskrbi paket hector_slam s hitrostjo 2 Hz. Algoritem brez načrtane poti, zemljevida in podatkov o poziciji ne more delovati. Naš algoritem temelji na naslednjih korakih.



Slika 29: Določitev lokalne poti in globalni cilj v prostoru.

1. Najprej glede na načrtano pot poiščemo najbližjo točko na dani poti od trenutne lokacije robota, tako lahko tvorimo novo lokalno pot robota, ki je sestavljena le iz nekaj točk ali določene nespremenljive razdalje. Trenutna pozicija robota je $X_s(x_x, y_s)$ in lokalni cilj je označen z $X_{lg\ (x_g, y_g)}$. Lokalna pot je določena z razdaljo pogleda vnaprej (Pvn). Dolžina pogleda vnaprej je določena glede na maksimalno hitrost robota in določeno časovno obdobje, ki ga lahko uporabnik spreminja. Dolžina parametra Pvn je enaka parametru W (glej sliko 30). Ta prikazuje določeno pot, kjer je s lokalnim ciljem določena lokalna pot, kateri se uporabi za izračun ujemajočega krožnega loka.

2. Algoritem temelji na krožnem gibanju, zato potrebujemo približek krožnice, ki jo tvorijo točke na lokalni poti. Za izračun radija R po enačbi 29 potrebujemo širino in višino trikotnika. Širina je razdalja med trenutno lokacijo in lokalnim ciljem, definirana s parametrom W, višino pa izračunamo s pomočjo trikotnikov, ki se tvorijo s tremi točkami – lokacija robota, ciljna točka in odklon – to so vmesne točke, trikotnik lahko vidimo na sliki 30. Med temi trikotniki izberemo tistega z največjim odklonom in tako določimo višino H. Za izračun višine h uporabimo teorem površine trikotnika. Na podlagi teh dveh parametrov dobimo radij krožnice R, ki predstavlja želeno trajektorijo. Enačba 32 izhaja iz Heuronove formule o površinskem teoremu.

$$R = \frac{h}{2} + \frac{W^2}{8h} \quad , \tag{29}$$

kjer sta parametra h in W izračunana z:

$$W = \sqrt{(x_s - x_g)^2 + (y_s - y_g)^2}; W \geq 0 \quad (30)$$

$$h = \max_n(\frac{An*2}{sC}); h \geq 0 \quad (31)$$

$$An = \sqrt{ss * (ss - sA) * (ss - sB) * (ss - sC)} \quad (32)$$

$$ss = \frac{(sA+sB+sC)}{2} \quad (33)$$

Izpeljava enačbe (29):

$$R = h + h_r \quad (34)$$

$$A_r = \frac{1}{2}Wh_r \rightarrow h_r = \frac{2*A_r}{w}; h_r \geq 0 \quad (35)$$

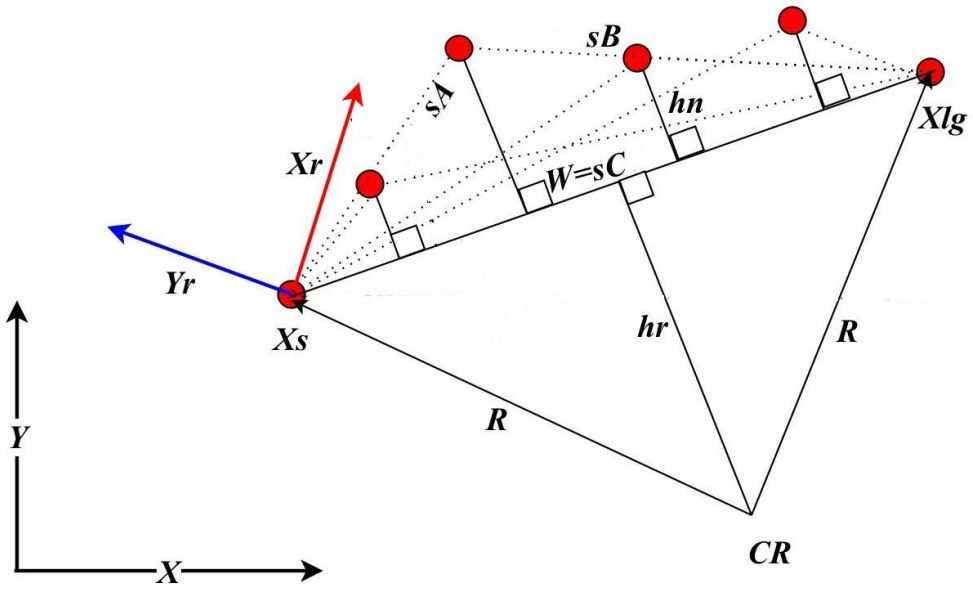
$$A_r = \frac{W}{4}\sqrt{4R^2 - W^2} \quad (36)$$

Vstavimo enačbi (35) in (36) v (34) in dobimo:

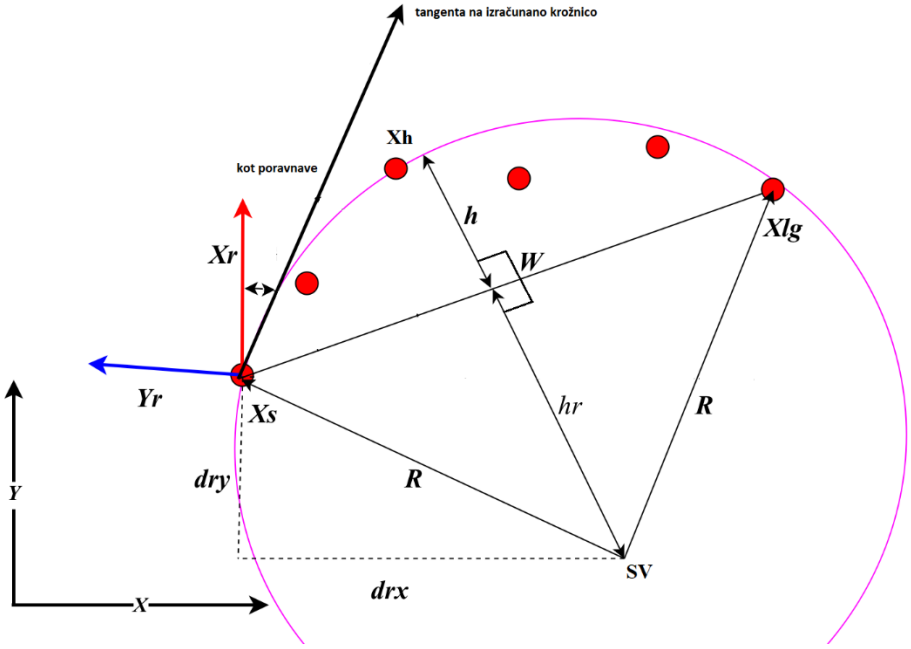
$$R = \frac{2*A_r}{w} + h = \frac{2*\frac{W}{4}\sqrt{4R^2-W^2}}{w} + h = \frac{h}{2} + \frac{W^2}{8h} \quad (37)$$

Enačbo izračuna krožnice radija kroga R izpeljemo s seštevkom višin dveh trikotnikov, ki sta tvorjena s tremi podanimi točkami in središčem iskanega kroga, kjer je h_n višina notranjega trikotnika in A_n ploščina notranjega kotnika (glej sliko 30), izračunana s Heronovo formulo za enakokraki trikotnik.

Slika 30 prikazuje trikotnike, ki se tvorijo na lokalni poti z namenom, da pridobimo odklon od najbližje poti do lokalnega cilja. Spremenljivke sA, sB in sC predstavljajo dolžine stranic, določene so s trenutnim položajem robota, lokalno ciljno točko in s tretjo točko na lokalni poti ter tvorijo trikotnik. Tretja točka na lokalni poti je določena na podlagi parametra pogleda vnaprej. Slika 31 prikazuje lokalno pot s točkami na tej poti in primer tvorjenega trikotnika, s katerim se določa višina odklona in s tem tudi približek krožnice. Ko dobimo vrednost H za izračun krožnice, uporabimo polovično vrednost, saj tako naredimo pot bolj gladko.



Slika 30: Določitev parametrov na lokalni poti.



Slika 31: Izračun največjega odklona in centra krožnega gibanja robota.

3. V naslednjem koraku moramo določiti še center krožnice na podlagi enačb (38) in (39), izračunana krožnica seka robotovo trenutno pozicijo Xs in njegov lokalni cilj Xlg in višino izračunanega trikotnika $Xh(x_h, y_h)$.

$$Cx = \frac{F1(y_g - y_s) - F2(y_h - y_s)}{2(x_h - x_s)(y_g - y_s) - 2(x_g - x_s)(y_h - y_s)} \tag{38}$$

$$Cy = \frac{F2(x_h - x_s) - F1(x_g - x_s)}{2(x_h - x_s)(y_g - y_s) - 2(x_g - x_s)(y_h - y_s)}$$

$$F1 = {x_h}^2 - {x_s}^2 + {y_h}^2 - {y_s}^2 \tag{39}$$

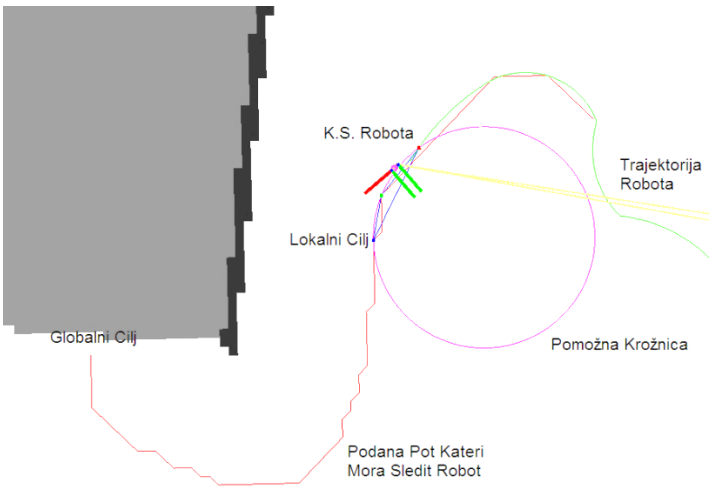
$$F2 = {x_g}^2 - {x_s}^2 + {y_g}^2 - {y_s}^2$$

V enačbi C_x in C_y predstavljata center krožnice, x_s in x_{lg} sta x koordinati začetne in končne točke lokalne poti, podobno velja za y os. Spremenljivka dir je lahko 1 ali –1, s katero določimo orientacijo krožnice. Linearna hitrost je določena in podana, ter je s strani uporabnika uravnavana kot maksimalna dovoljena hitrost. Hitrost izračuna krmilnih ukazov za gibanje robota je mogoče nastaviti za posamezne robote.

$$v = R * \omega \tag{40}$$

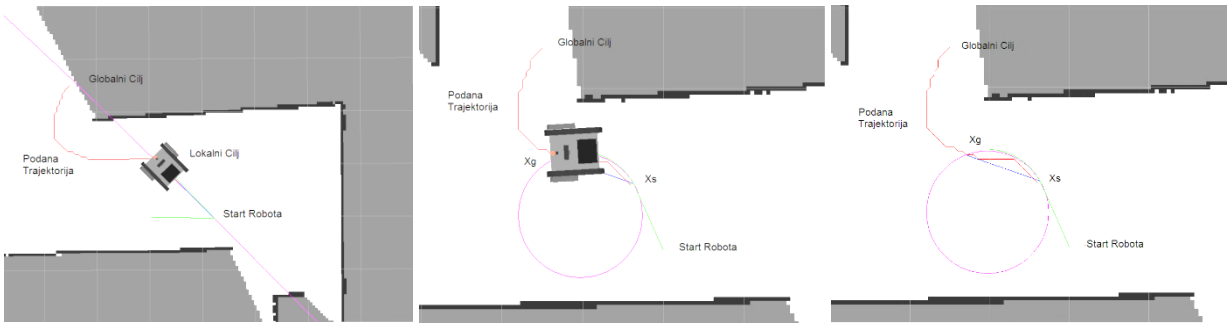
$$\omega = v/R \tag{41}$$

4. Spremenljivka $v\ [m/s]$ predstavlja linearno hitrost in $\omega\ [rad/s]$, kotno hitrost. V primeru, da ni premika robota, tudi če je hitrost podana, se linearna hitrost zvišuje do maksimalne dovoljene kotne hitrost pa se spreminja zvezno glede na enačbi (40) in (41). Izračun nove lokalne poti se izvede ob vsaki prispeli posodobitvi lokacije robota, s tem želimo dobiti čim bolj aktualne krmilne hitrosti za robota.



Slika 32: Primer izračuna in želene lokalne poti ter zgodovina sledenja.

Generirane hitrosti se direktno pošljejo bazi robota in se tako izvedejo, regulacija oziroma popravljanje poti je izvedeno ob vsakem prispelem podatku o položaju, seveda je pa to pogojeno z natančnostjo informacije. V našem primeru porabljamo hector_mapping, ki dvakrat na sekundo priskrbi informacijo o lokaciji robota, kar zadostuje za natančno vodenje robota po želeni trajektoriji. Slika 32 prikazuje primer simulacije, rdeča črta predstavlja načrtano pot, zelena črta že prevoženo pot, vijoličasta in modra sta pomožni črti za izračun hitrosti, ter bazni koordinatni sistem robota (K. S. Robota) z rdečo X osjo, orientirano v smeri krožnice. Ko robot doseže globalni cilj, čaka na novi cilj. Priporočljivo je, da je toleranca kota čim manjša, saj tako dosežemo zelo dobro prileganje trajektoriji.



Slika 33: Primeri izračuna pomožne krožnice.

Rdeče črte predstavljajo podano pot, na podlagi katere je določena trenutna lokalna pot, začetek in konec je označen z modro črto, konec se nahaja na dani poti. Z najbolj oddaljeno točko od središčnice na lokalni poti je tvorjen trikotnik, na podlagi katerega je izračunan radij krožnice, ki je označena z vijoličasto bravo. V prvem primeru, slika 33 – levo, vidimo, da je radij krožnice zelo velik, saj je odsek krožnice na sliki praktično raven. V drugem primeru, slika 33 – sredina, vidimo robota z izračunano krožnico z radijem, ki pokriva trenutno podano pot. Slika 33 – desno pa prikazuje pot brez vrisanega robota.

Tabela 2: Tabela parametrov za nastavljanje algoritma za sledenje trajektoriji

Št:	**Parameter:**	**Vrednost:**	**Tip:**	**Opis:**
1.	pub_cmd_hz	10	double	Frekvenca, s katero se vrednosti linearne in kotne hitrosti pošiljajo, vrednost je podana v Hz.
2.	path_topic	/exploration_path	string	Tema, ki predstavlja načrtano pot v prostoru in je določena z nav_msgs/Path tipom s točkami (x,y,z) in rotacijo po oseh (r, p, y) v koordinatnem sistemu robota.
3.	sub_pose_update_topic	/odom	string	Tema, ki poda trenutno lokacijo robota v koordinatnem sistemu robota (x,y,z, r, p, y).

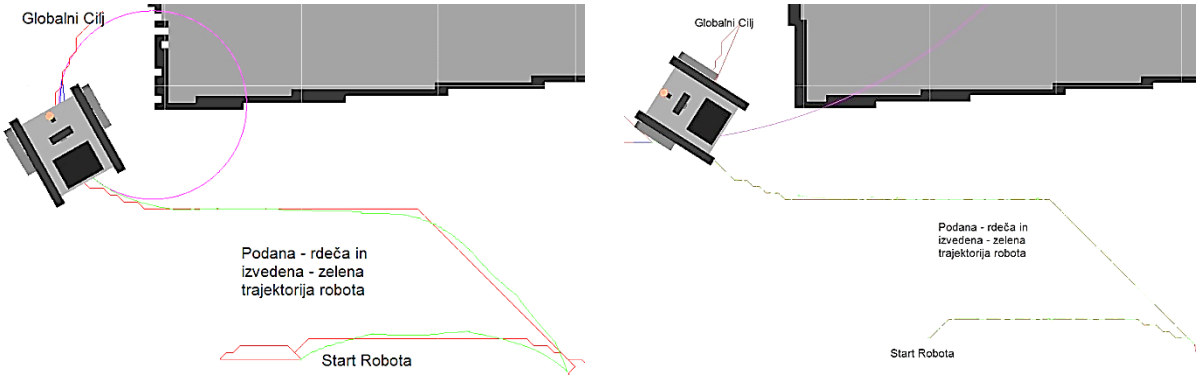
4.	out_cmd_vel	/cmd_vel	string	Ime teme, na katero so hitrosti objavljene.
5.	robot_trajectory	/trajectory	string	Ime teme, na katero je objavljena trajektorija robota.
6.	map_link	/map	string	Bazni koordinatni sistem.
7.	base_link	/base_footprint	string	Robotov bazni koordinatni sistem.
8.	max_lin_speed	/0.3	double	Omejitev maksimalne linearne hitrosti robota v m/s.
9.	min_lin_speed	/0.1	double	Omejitev minimalne linearne hitrosti robota v m/s.
10.	max_rot_speed	/0.5	double	Omejitev maksimalne kotna hitrosti robota v rad/s.
11.	min_rot_speed	/1	double	Omejitev minimalne kotne hitrosti robota v rad/s.
12.	rot_correction_factor	1.0	double	Faktor, ki je pomnožen z izračunano kotno hitrostjo, to uporabljamo zaradi različnih baz in različnih obnašanj le-teh na kotno hitrostno krmiljenje.
13.	execute_periode	1.0	double	Faktor, podan v sekundah, določa pot na trajektoriji, glede na maksimalno možno hitrost, ta je izračunana ob vsakem prihodu nove raziskovalne poti.
14.	update_skip_untile_vel_ increase	5	double	Faktor, ki določa, kako dolgo se linearna hitrost ne povečuje, sistem ne dviguje linearne hitrosti, da se reši morebitnih zagozditev.
15.	global_goal_tolerance	0.1	double	Razdaja, podana v metrih, določa natančnost, katera mora biti pri doseganju globalnega cilja
16.	angle_correction	0.4	double	Faktor, podan v radianih, določa natančnost poravnave robota glede na kot izračunane tangente, manjši je faktor, bolj se robot prilagaja podani trajektoriji.

Parametri, opisani v Tabeli 1, določajo vedenje robota pri sledenju podani trajektoriji. Slednjega lahko prilagodimo s spreminjanjem vrednosti parametrov. Z njimi lahko na končni rezultat vplivamo tako, da se trajektorija robota bolj prilega dani trajektoriji. Prvi primer na sliki 34 – levo je prikazan s parametroma execution_period in angle_correction, ki najbolj vplivata na zgled končne trajektorije:

- execution_period: 2.0
- angle_correction: 0.4

V drugem primeru, slika 34 desno, smo spremenili parametre na:

- execution_period: 0.5
- angle_correction: 0.1

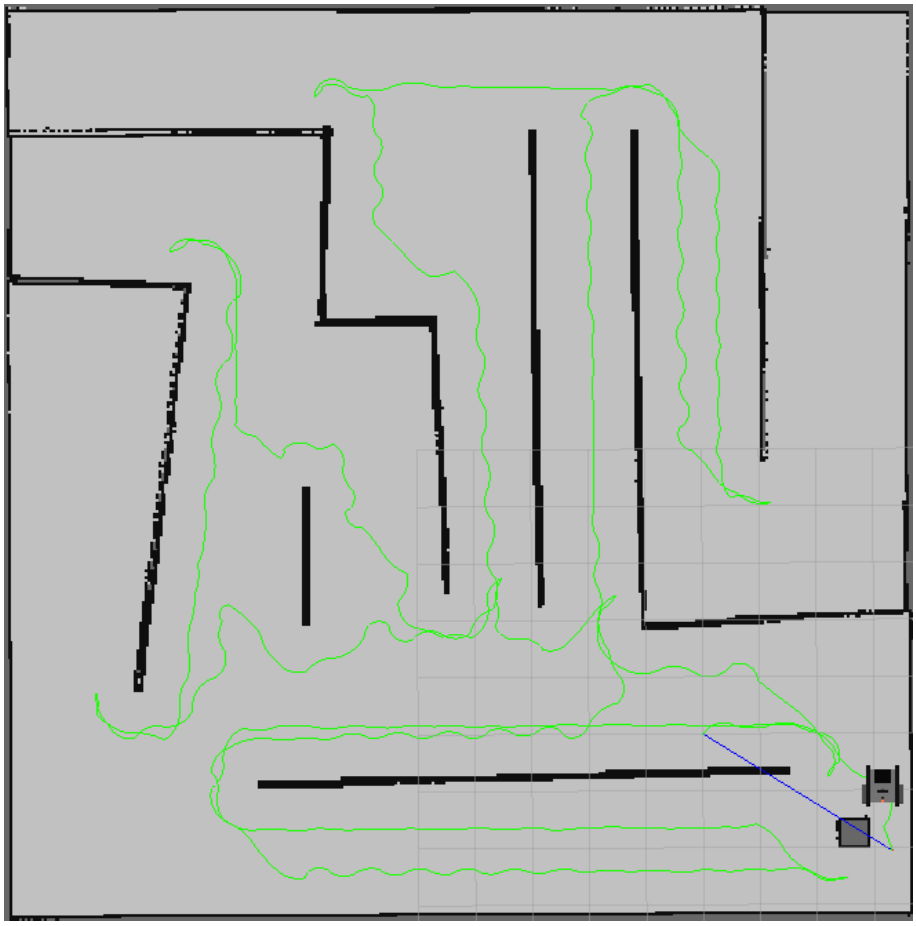


Slika 34: Primerjava sledenja poti.

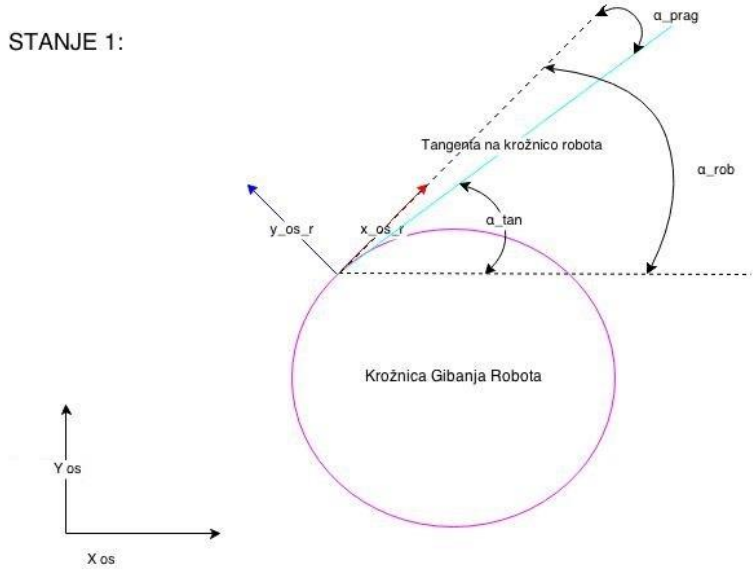
Kot lahko vidimo na sliki 34, je načrtana pot v zeleni barvi na levi strani manj podobna rdeče načrtani poti na desni. Vzrok temu je povečan faktor angle_correction, ki določa, koliko lahko kot pri potovanju robota odstopa od pravega kota, ki je idealno poravnan s trajektorijo. Parameter execution_periode pa daje robotu razdaljo pred njega, kar pomeni, da s tem parametrom povečujemo ali zmanjšujemo radije, ki jih robot izvede na zavojih, kot je prikazano sliki 34 – desno. Natančnost izvedbe je seveda pogojena s hitrostjo; čim bolje robot sledi podani poti, dalj časa potrebuje, da prevozi isto razdaljo. Točke na poti, ki so pridobljene iz planerja poti Hector, niso zvezne in ne tvorijo primitivnih likov, in tudi niso enakomerno porazdeljene v prostoru, kar moramo upoštevati pri sledenju poti.

Algoritem je treba prilagoditi robotski bazi, na kateri se izvaja sledenje poti. Pred izboljšavami je bilo delovanja algoritma precej robustno, kot je mogoče videti na sliki 35.

Zaradi robustnega izgleda rezultirajoče trajektorije smo se odločili, da dodamo izboljšavo. Izboljšava temelji na kompenzaciji krožnega gibanja med kombinacijo linearnega in krožnega gibanja. To pomeni, da krožno gibanje med premikom prilagajamo glede na trenutni kot robota. To prinese bolj gladko trajektorijo in manj zaustavljanja robota na poti, kar pomeni, da je robot hitrejši pri sledenju poti. Sedaj imamo tri stanja robota, ki so opisana v nadaljevanju.



Slika 35: Delovanje algoritma brez dodatnih funkcij.



Slika 36: Primer določanja stanj robota za kompenzacijo kota, primer stanja 1.

1. **STANJE 1:** Robot je glede na tangento krožnice poravnan znotraj določenega pragu. V tem primeru se izračunajo hitrosti glede na trenutno pozicijo in se izvedejo, kot je vidno na sliki 36.

2. **STANJE 2:** Robot je glede na tangento in prag odmaknjen, vendar še vedno v drugem pragovnem pasu. V tem primeru trenutno orientacijo robota kompenziramo med premikom robota tako, da dodamo kotno hitrost (42), ki je izračunana v enačbi (43).

$$rot_{hit} = \omega = \omega - \omega_d \tag{42}$$

$$\omega_d = (\alpha_{tan} - \alpha_{rob})/p \tag{43}$$

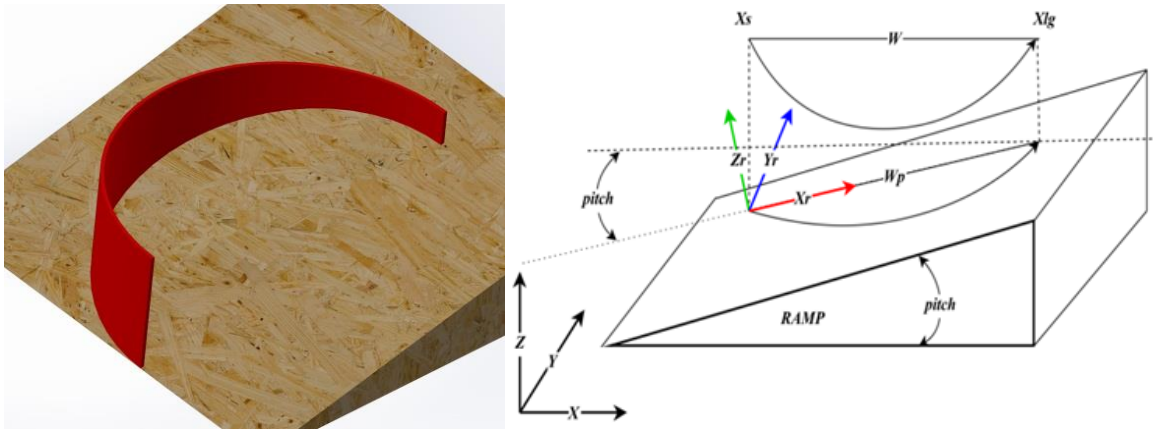
Kjer je ω_d dodatna kotna hitrost za kompenzacijo kota robota, α_{tan} je kot tangente na krožnico robota in α_{rob} predstavlja trenutno orientacijo robota. Časovna perioda je določena s p.

3. **STANJE 3:** Robotova orientacija je nad obema pragovoma in je ne kompenziramo med premikanjem, to pomeni, da mora robot izvesti vrtenje na mestu.

Rezultat kompenziranega obnašanja algoritma s kompenzacijo kotov je mogoče videti na sliki 40.

Algoritem za sledenje poti smo preizkusili tudi v realnih okoliščinah z robotom, imeli smo težavo z zamikom robota med rotacijo. Robot se zaradi nepredvidenih sil trenja v prostoru ni obračal točno tako, kot je bilo pričakovano. To je bilo opazno tudi na klancih, kjer je bil ta zamik precej izrazitejši.

Potovanje robota po klancih omejuje tudi projekcija krožnic na ravnino, ta tvori rahlo spremenjene krožnice oziroma elipse, ki so lahko na manjših klancih zanemarljive. Primer projekcije krožnice na klanec je na sliki 37.



Slika 37: Projekcija krožnice na klanec.

Problem drsenja na mestu v primeru rotacije smo omilili z vpeljavo zelo male krožnice, ki predstavlja rotacijo na mestu, tako smo skušali kompenzirati napako pri rotaciji robota.

Na spremembo lokalne trajektorije na klancu vpliva višina klanca. Krožnica se spremeni v elipso, ena stranica se podaljša v smeri robota, to lahko izračunamo z enačbo (44), seveda če imamo na voljo informacije o zasuku robota. Kjer je dRx razteg ravnine krožnice in α trenutni naklon robota. Trenutno rotacijo robota lahko pridobimo iz inercialnih enot. Rezultat potovanja po klancu je, da z obstoječim postopkom ne dosežemo pravega lokalnega cilja, saj se krožnica s projekcijo na klanec raztegne. Pričakovana nova pozicija robota je $R + dR$, vendar zaradi klanca robot v teoriji obstane na razdalji rRx. Zaradi zelo majhnih razlik lahko ta zamik tudi zanemarimo, predvsem ker se trajektorija računa hitreje, kot robot dosega lokalni cilj, robot izračuna novo lokalno trajektorijo glede na svojo trenutno pozicijo ob vsaki pridobitvi položaja robota.

$$dRx = (\frac{W}{\cos(\alpha)}) - W \tag{44}$$

$$rRx = \mathrm{W} - \cos(\alpha) * dRx \tag{45}$$

Podobno velja za drugo os glede na ravnino robota, kjer se spreminja druga dimenzija višine krožnice h:

$$dRy = (\frac{h}{\cos(\beta)}) \tag{46}$$

$$rRy = \mathrm{h} - \cos(\beta) * dRy \tag{47}$$

V primeru, da želimo upoštevati kompenzacijo poti do danega lokalnega cilja, moramo narediti nov izračun krožnice z novimi parametri:

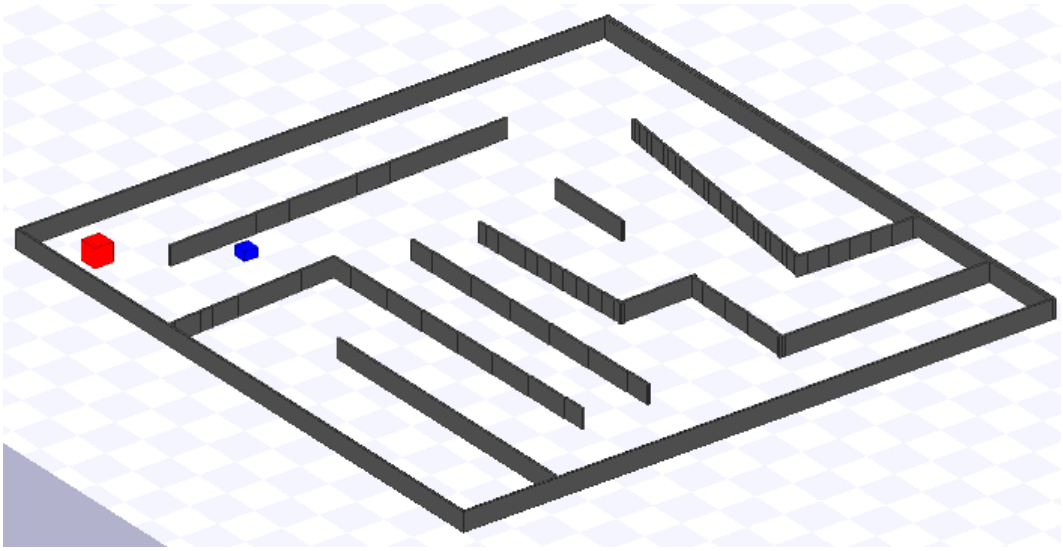
$$W = W + abs(dRx) + abs(dRy) \text{ in } h = h \tag{48}$$

Tako dobimo novo krožnico, ki kompenzira kot naklona robota. Na podlagi te se izračunajo nove krmilne hitrosti.

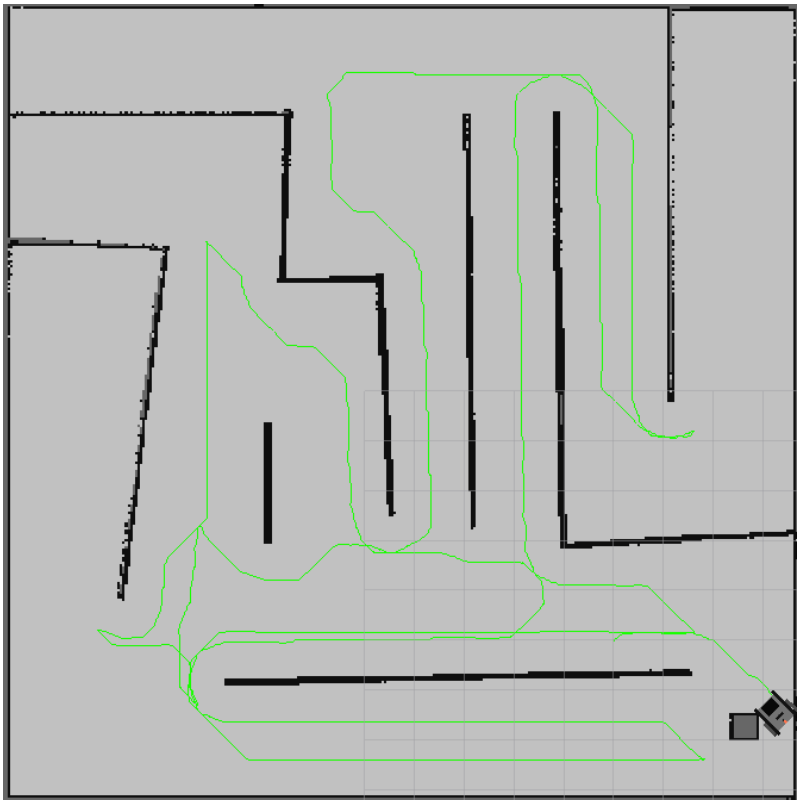
Potovanje mobilnega kolesnega ali goseničnega robota po klancih ali stopnicah ima nekaj posebnosti, ki jih je treba upoštevati. Takšen mobilni robot pri premagovanju stopnic potrebuje posebne pogoje, da uspešno premaga stopnice. Zaželeno je, da je robot pred vstopom na stopnice paralelno poravnan in da je med premagovanjem stopnic čim bolj ravna trajektorija, da ne prihaja do zdrsov.

4.2 SIMULACIJSKI EKSPERIMENT

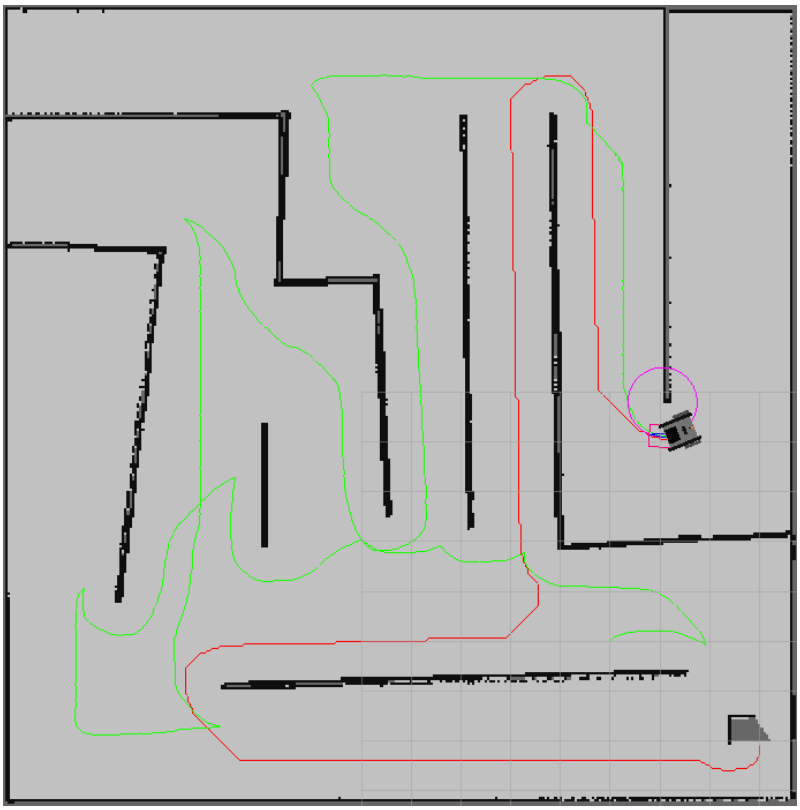
Robot v avtonomnem načinu raziskuje samo prostore, ki jih lahko prečka oziroma prevozi. Če je luknja v steni premajhna za njegovo bazo, tega prostora ne bo raziskal, kar povzroča težave predvsem, če laserski merilnik naredi napako na zemljevidu. Na sliki 38 vidimo simulirano okolje v stopenjskem (angl. *stage*) simulatorju. Robot je okolje avtonomno raziskal z orodji Hector. ROS stage simulacijsko okolje predstavlja enostavno in odzivno simulacijsko orodje v ROS. Na sliki 39 lahko vidimo, da je trajektorija rahlo nazobčana in na določenih mestih pride do težav pri poravnavi, kar povzroči gibanje robota naprej in nazaj v določeni točki. Slika 40 pa prikazuje okolje, raziskano z algoritmom DPKL. Opazimo lahko bolj gladko trajektorijo brez zobov tam. Uporabili smo podoben algoritem za upravljanje raziskovanja, kot ga uporablja Hector. Opazi se tudi, da je trajektorija, narejena z algoritmom DPKL, bolj optimalna kot s pristopom Hector.



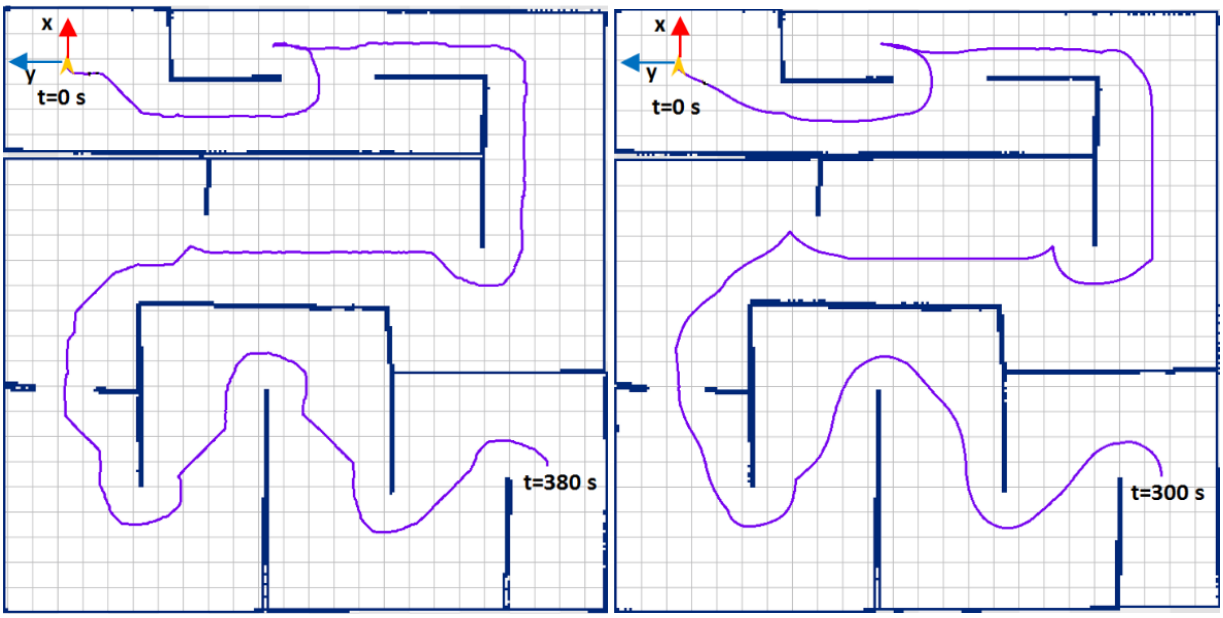
Slika 38: Simulacijsko okolje ROS Stage.



Slika 39: Avtonomno preiskovanje prostora z orodjem Hector.

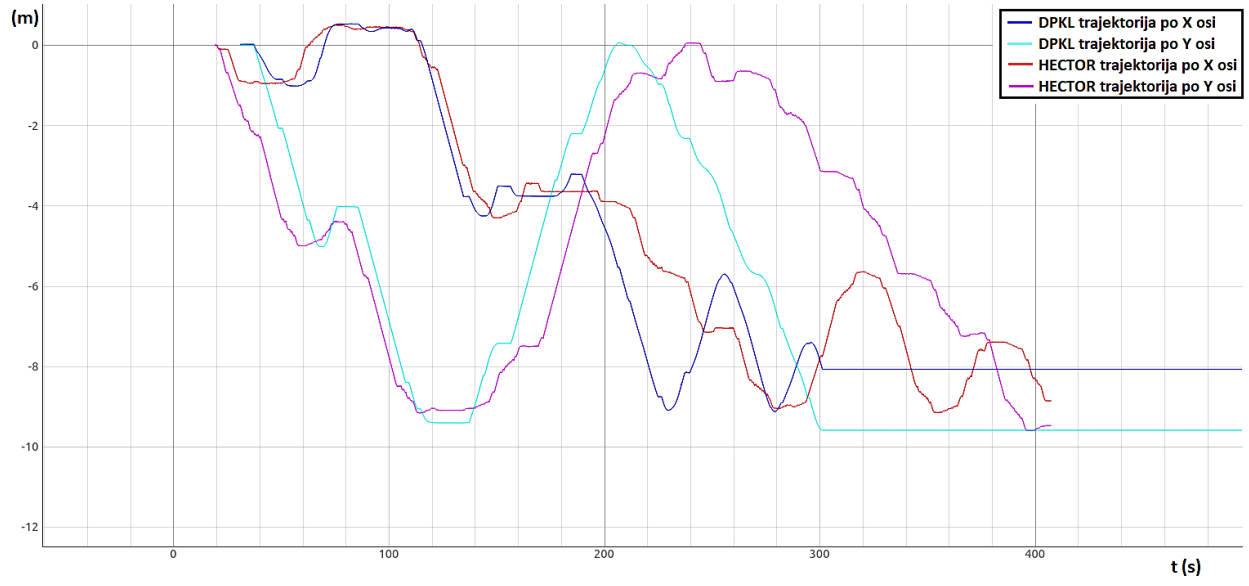


Slika 40: Avtonomno preiskovanje z algoritmom DPKL.

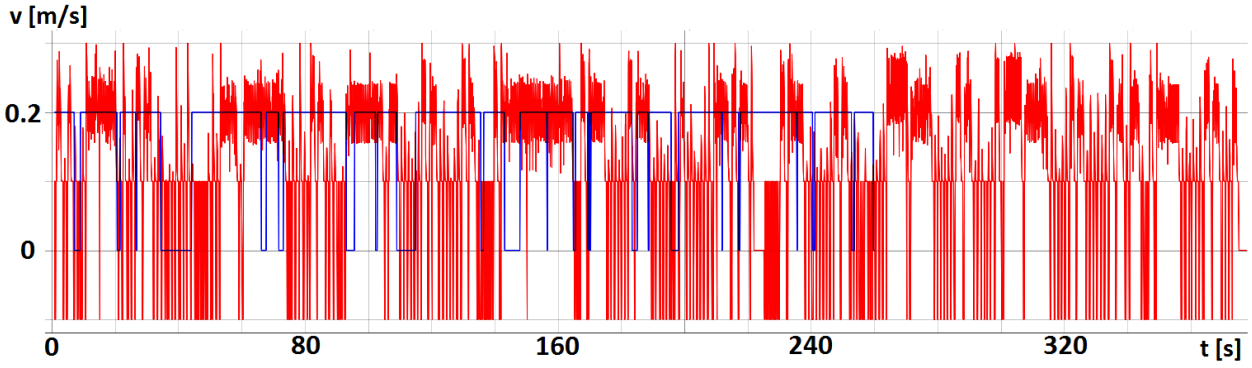


Slika 41: Primerjava simulacijskih rezultatov med Hector path follower in DPKL.

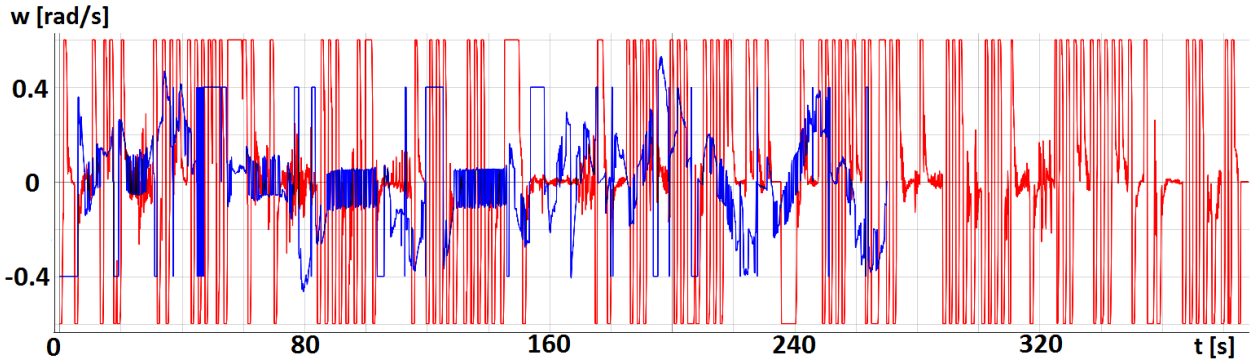
Opazimo lahko, da je algoritem Hector ustvaril bolj nazobčano in manj gladko pot v prostoru. Medtem ko je DPKL izdelal precej gladko pot, nazobčana je tam, kjer je robot dosegel svoj cilj in nato nadaljeval z raziskovanjem prostora. Očitna razlika pa je vidna v porabljenem času za izvajanje algoritmov.



Slika 42: Primerjava pozicije robota skozi čas med Tedusar in Hector.



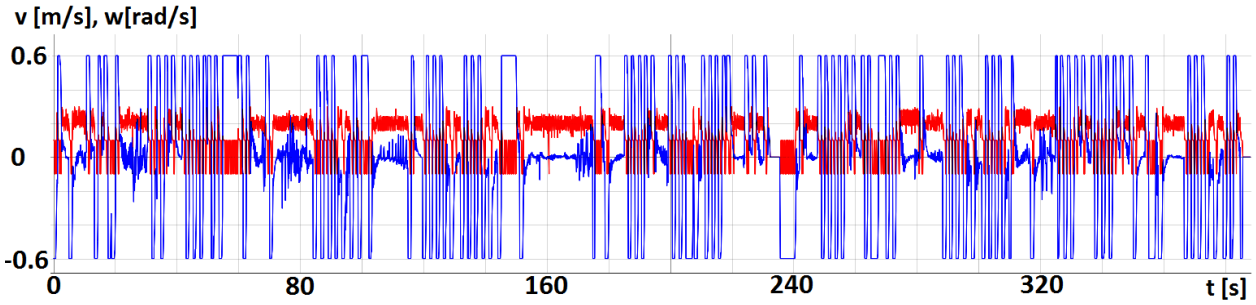
Slika 43: Primerjava podane linearne hitrosti med DPKL (modra) in Hector (rdeča), v odvisnosti od časa.



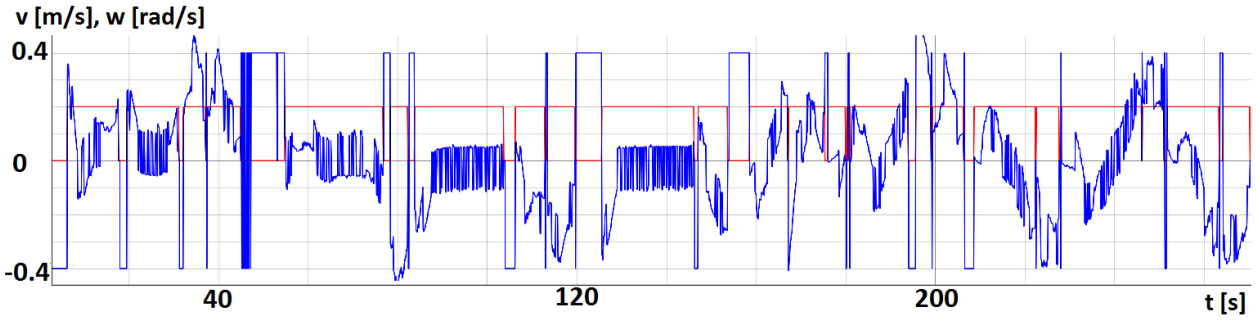
Slika 44: Primerjava kotne hitrosti med DPKL (modra) in Hector (rdeča), v odvisnosti od časa.

Graf na sliki 42 prikazuje položaj robota po času, na X in Y osi, kjer modri (X) in turkizen (Y) graf pripadata algoritmu DPKL, rdeči (X) in vijoličast (Y) pa Hectorju. Opazimo lahko podobnost krivulj, ki so različno glajene, v primeru algoritma DPKL je manj zobov in tudi cilj je dosežen hitreje. Sliki 43 in 44 prikazujeta časovno odvisnost hitrosti, modra barva predstavlja hitrosti algoritma DPKL in rdeča hitrosti, proizvedene z algoritmom Hector. Pri primerjavi linearne hitrosti lahko vidimo, da je v primeru DPKL sprememba hitrosti precej manjša, hitrost je bolj konstantna kot pri algoritmu Hector, prav tako lahko vidimo, da Hector v veliko primerih pelje z negativno linearno hitrostjo, torej nazaj. Vidimo tudi, da pri vožnji naprej algoritem Hector prilagaja linearno hitrost, ki doseže maksimalno 0,3 m/s. Algoritem DPKL v tem primeru doseže maksimalno hitrost 0,2 m/s in je glede na končen rezultat hitrejši od Hectorja. Slika 45 prikazuje linearno (rdeča barva) in kotno (modra barva) hitrost, ki je izračunana v odvisnosti od časa z algoritmom Hector. Vidimo lahko povezavo med linearno in kotno hitrostjo, kjer je opazna velika oscilacija kotne hitrosti. Slika 46 prikazuje linearno (rdeča barva) in kotno (modra barva) hitrost, ki je izračunana z algoritmom DPKL . V tem primeru lahko opazimo konstantno linearno in spremenljivo kotno hitrost, ki variira v določenih pragovnih območjih. Opazimo tudi, da tudi kotne hitrosti dosegajo večja nihanja kot v

primeru DPKL , vendar so veliko bolj razgibane, robot v tem primeru veliko oscilira, torej se premika na mestu levo in desno, s tem izgublja čas in energijo. V primeru kotnih hitrosti DPKL opazimo, da so krmilne kotne hitrosti bolj zvezne, vendar še vedno oscilirajo v manjšem področju, ki je definirano s spodnjim pragom kotne hitrosti. Maksimalna kotna hitrost je podana v primeru rotacije na mestu, kjer lahko vidimo, da je v tistem trenutku linearna hitrost enaka nič.



Slika 45: Linearna (rdeča) in kotna (modra) hitrost, izračunana z algoritmom Hector, v odvisnosti od časa.

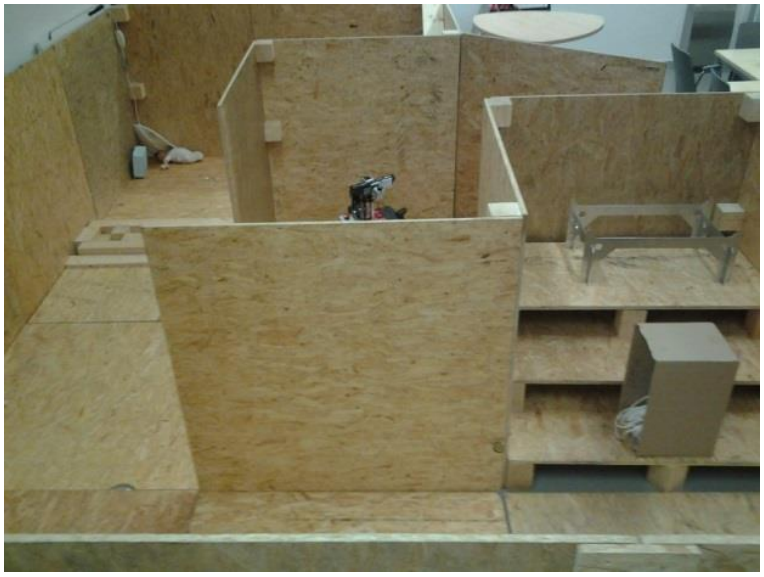


Slika 46: Linearna (rdeča) in kotna (modra) hitrost, izračunana z algoritmom Tedusar, v odvisnosti od časa.

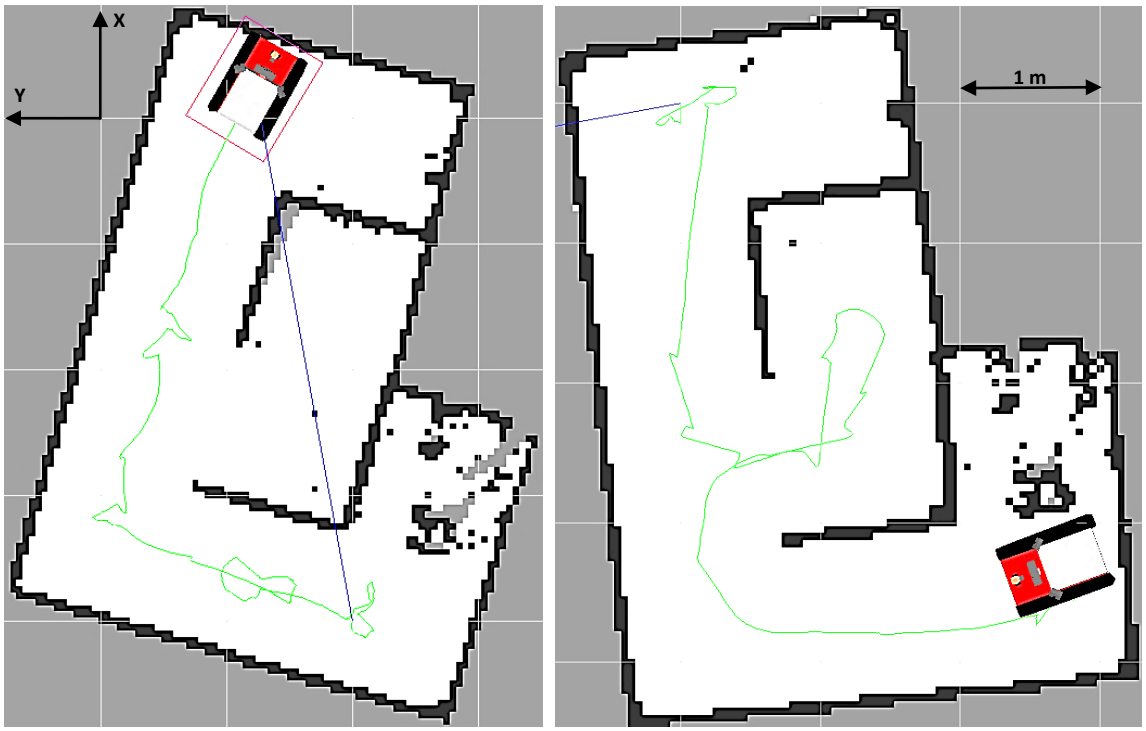
Opisali smo del celote, ki predstavlja avtonomni robotski sistem za preiskovanje prostora, osredinili smo se na sledenje poti in upravljavca raziskovanja. Algoritem za preiskovanje in izdelovanje zemljevida smo uporabili iz programskega paketa Hector. Pri izvedbi programa smo upoštevali čim večjo prilagodljivost algoritma različnim robotskim bazam, dokazali smo tudi bolj optimalno in manj nazobčano pot v primerjavi s programskim paketom Hector. S tem delom smo želeli narediti izvedljiv program, ki bo lahko varno in zanesljivo vodil naše pakete v prostorih. Pri tem smo upoštevali čim bolj fleksibilno in prilagodljivo izvedbo programa. Za zaključitev celote smo morali dodati tudi paket, ki upravlja preiskovanje, saj z zagonom omenjenih paketov dobimo avtonomni sistem preiskovanja.

4.3 REALNI EKSPERIMENTI

Realni eksperimenti so bili izvedeni v standardni NIST areni [7, 56], ki se tudi uporablja na RoboCup tekmovanjih [57].

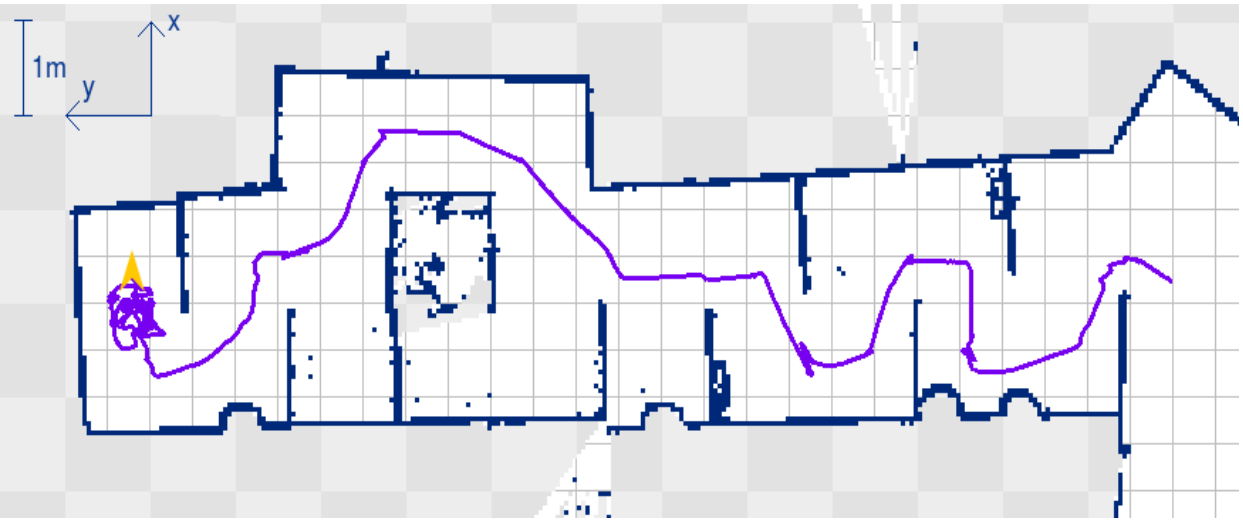


Slika 47: Primer arene, v kateri je potekalo avtonomno raziskovanje.

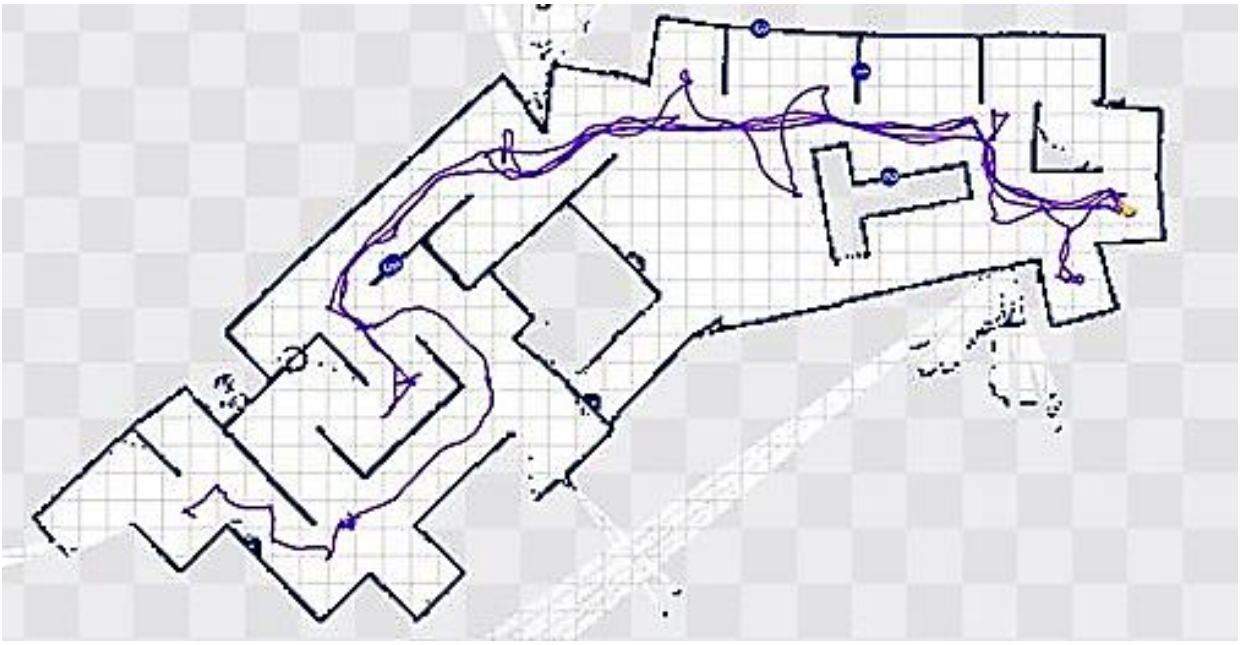


Slika 48: Desno raziskovanje arene s teleoperiranjem in levo raziskovanje arene v avtonomnem načinu delovanja.

Raziskovanje arene s teleoperiranjem smo naredili zato, da lahko rezultat avtonomnega raziskovanja primerjamo z daljinsko upravljanim. Na sliki 48 – desno lahko opazimo, da je zemljevid v primeru teleoperianja nekoliko boljši, saj smo naredili več poti kot takrat, ko je robot sam prevozil areno. Opazimo lahko tudi negladko trajektorijo robota v avtonomnem načinu, do tega pride predvsem zaradi obračanja robota, saj pri vrtenju robot ne ostane na isti točki, ampak se premika, zato se pojavijo izbokline na trajektoriji. Robot v avtonomnem načinu raziskuje samo prostore, katere lahko prečka oziroma prevozi.

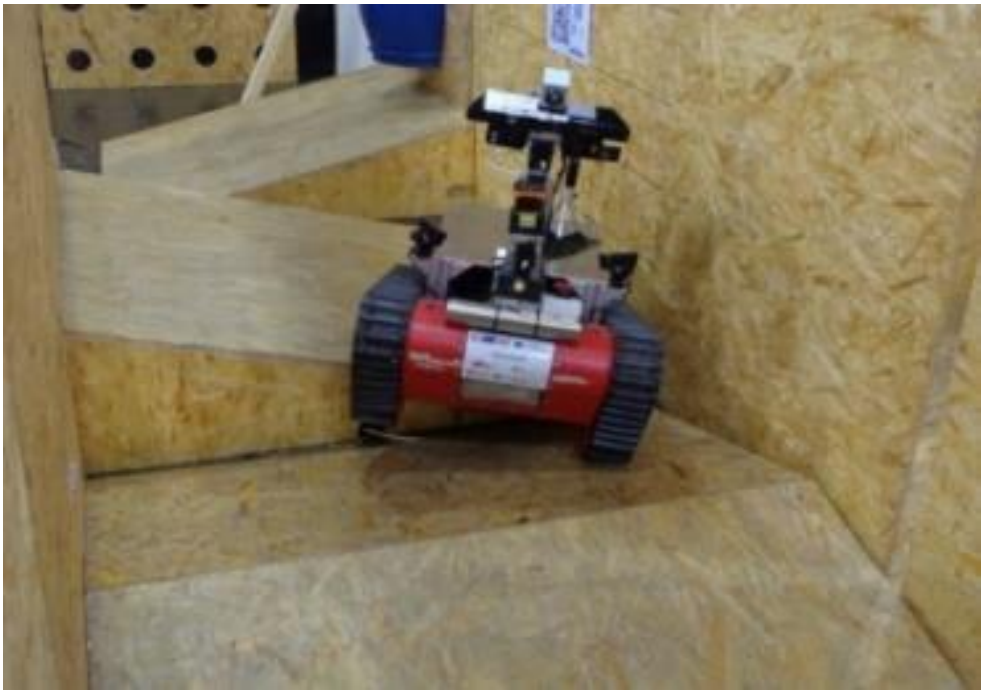


Slika 49: Del zemljevida, izdelan na tekmovanju German Open 2013.



Slika 50: Zemljevid, izdelan na tekmovanju RoboCup 2014 v Braziliji.

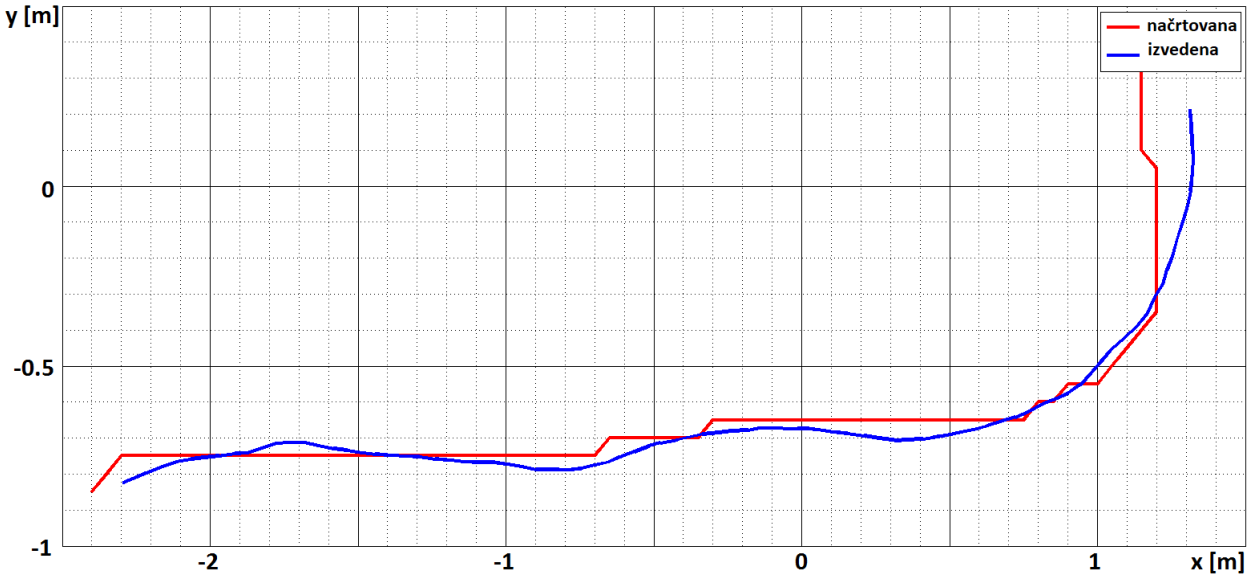
Algoritem za sledenje poti je bil testiran na tekmovanju RoboCup German Open, kjer so roboti tekmovali v standardni NIST areni. Na našem prvem tekmovanju smo za sledenje poti uporabljali move_base, to je standardni paket za navigacijo v ROS-u. Vendar naš robot ni uspešno sledil začrtani poti. Sledenje poti smo poizkusili tudi z Hector path follower, ki prav tako ni bil uspešen v realnem eksperimentu. Ni nam uspelo prilagoditi parametrov v takšni obliki, da bi dosegli zanesljivo sledenje, saj je robot večkrat trčil ob stene arene. Na tekmovanju German Open 2013 smo uporabili lasten algoritem za sledenje; sledenje in izris zemljevida je mogoče videti na sliki 49 in sliki 50. Opazimo lahko, da je robot na začetku naredil nekaj zasukov, kar je povzročilo drsenje, predvsem zaradi neurejenega okolja. Neurejena tla na tekmovanju RoboCup German Open je mogoče videti na sliki 51. Med potovanjem se je robot nepredvidljivo premikal zaradi kompleksne sestave tal. Z našim algoritmom za sledenje smo uspešno kompenzirali napake pri vrtenju in zdrsu robota. Avtonomni mobilni robot je zanesljivo sledil zadani poti.



Slika 51: standardna arena NIST na tekmovanju RoboCup German Open 2013.

Naša naslednja večja preizkušnja je bila na svetovnem prvenstvu RoboCup 2014 v Braziliji. Slika 52 prikazuje izdelan zemljevid in trajektorijo robota. Z našim mobilnim robotom in algoritmom za sledenje smo uspeli izdelati največji zemljevid.

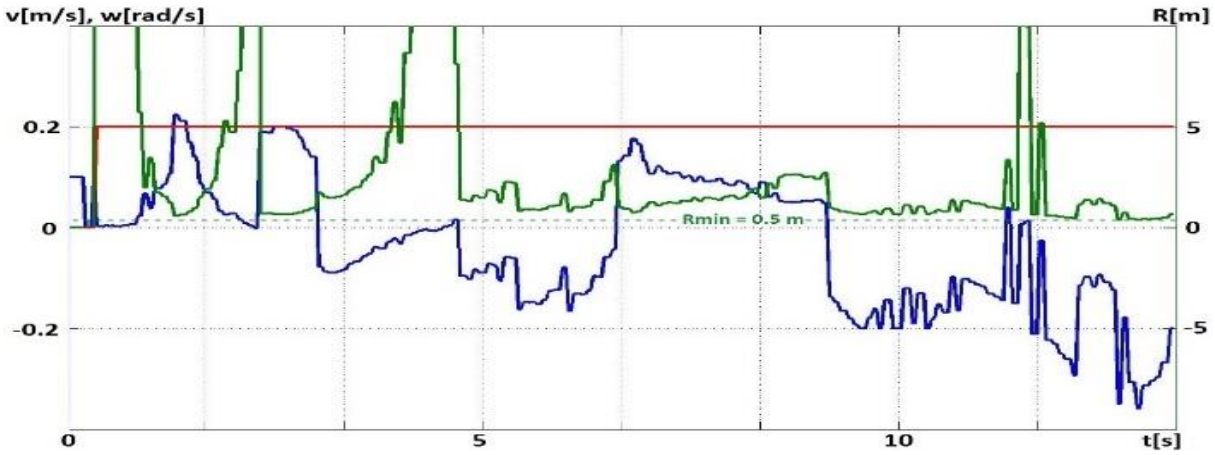
Slika 52: Celotna pot robota, rdeča - načrtovana pot, modra - izvedena pot.



Slika 53: Del izvedene poti v realnem eksperimentu.

Da bi ocenili delovanja algoritma za sledenje, smo posneli podatke med realnim eksperimentom na tekmovanju RoboCup 2014. Naš navigacijski sistem je zasnovan tako, da si izračuna novo pot v prostoru vsakih 15 sekund, z namenom raziskati čim večji prostor. Slika 53 prikazuje načrtovano pot, obarvano rdeče, in izvedeno pot, obarvano modro. Opazimo lahko večje zasuke robota, kar je posledica načrtovanja

novih poti v prostoru vsakih 15 sekund. Na sliki 53 bolj podrobneje prikažemo del izdelane poti. Tukaj prikazujemo približno 3,5 m dolg odsek, kjer je robot izvedel sledenje poti z maksimalno translacijsko hitrostjo 0,4 m/s in maksimalno kotno hitrostjo 0,2 rad/s. Dodatni parametri, Pvn: 0,8 m, an_th1: 0.15 rad in an_th2: 0.9 rad. Nastavitve parametrov so popolnoma enake, kot so bile na tekmovanju RoboCup 2014 v Braziliji.



Slika 54: Hitrosti sledenja, rdeča - translacijska hitrost, rdeča - kotna hitrost, zelena – izračunan radius R.

Modra črta na sliki 53 sledi rdeči črti. Tla v areni, kjer smo izvedli eksperiment, so bila gladka in drsljiva. Eksperiment pokaže, kako robot pri vrtenju oddrsi stran. Razlike med rdečo in modro črto so hitrostni pogreški, izvedbene napake, nepredvidljiva tla in nastavitve parametrov. Maksimalna napaka na osi X je 0,28 m in maksimalna napaka na osi Y je 0,12 m. Slika 54 prikazuje časovno odvisnost kotne in translacijske hitrosti. Tu lahko opazimo konstantno translacijsko hitrost in zelo aktivno kotno hitrost. Zelena črta prikazuje trenutno izračunan radij, ki je uporabljen za izračun translacijske in kotne hitrosti. Prav tako je viden minimalen izračunan R, kjer v maksimumu ni omejitev. Kljub vsem motnjam in razgibanim tlom je robot uspešno sledil začrtani poti.

Omenjeni algoritem je mogoče preizkusiti v simulacijskem okolju ali z realnim robotom. Algoritem DPKL smo naložil v skupnost ROS, kjer je dostopen vsem [59].

Hipoteza **H1** predvideva, da bo mobilni robot uspešno sledil podani poti s pomočjo dinamičnega prilagajanja krožnic potovanja. To delovanje in obnašanje DPKL algoritma smo dokazali z rezultati v simulacijah in realnih eksperimentih, prav tako smo ga primerjali s postopkom sledenja Hector. Algoritem za izračun kotne hitrosti uporablja iskanje krožnic, ki se ob vsaki spremembi položaja robota posodobijo. S pomočjo teoretično izračunane krožnice, preračuna hitrosti s katerimi želi izvesti izračunano krožnico. S

hitrim spreminjanjem kotne hitrosti omogoča mobilnemu robotu zanesljivo doseganje podanih točk v prostoru, skupaj z pomožnimi funkcijami. Na podlagi simulacijskih in realnih eksperimentov, lahko potrdimo hipotezo **H1**, saj je robot v realnih okoljih z kompleksno strukturo uspešno sledil podani poti.

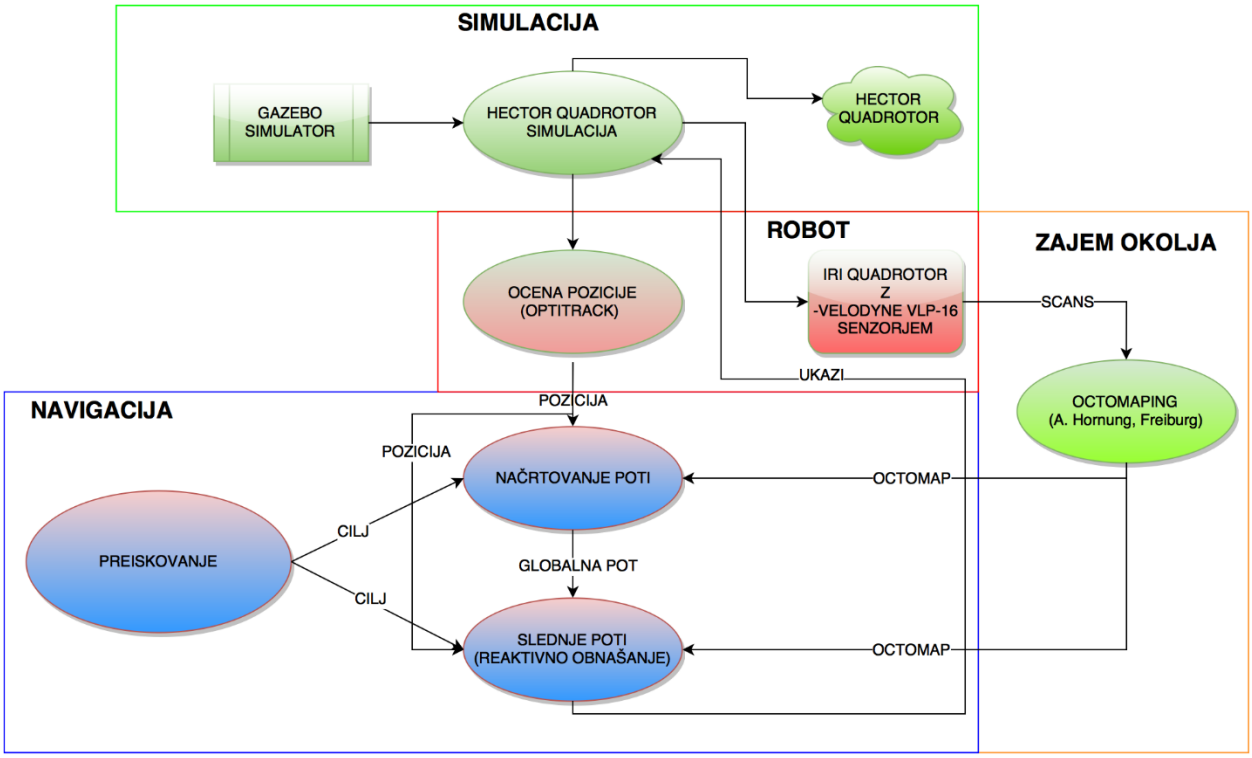
5 NAVIGACIJSKI SISTEMI LETEČIH MOBILNIH ROBOTOV

Do sedaj smo obravnavali sistema in del navigacijskega sistema, ki je zmožen z dano robotsko bazo slediti načrtani poti. Tukaj smo zaključili celoto z algoritmom DPKL , ki predstavlja navigacijski sistem za kopenske robote, ki se lahko gibajo samo po tleh. Kopenski mobilni roboti imajo omejeno mobilnost, kar pa je zelo pomemben faktor v kompleksnih okoljih, zato smo naše delo želeli razširiti še z robotom ki je zmožen leteti. Na to temo smo raziskali obstoječo literaturo in obstoječe algoritme v ROS-u. Naleteli smo na zelo malo rešitev, zato smo se odločili, da se sami lotimo razvoja navigacijskega algoritma za leteče robote. Podrobneje opisan algoritem DPKL smo prilagodili letečemu robotu. Podrobnejši opis s podanimi rezultati sledi v nadaljevanju tega poglavja in se navezuje na hipotezo (**H4**).

Ta del se osredotoča na gradnjo 3D-navigacijskega sistema za preiskovanje neurejenih področji s pomočjo BLV. BLV ima nalogo, da preišče področje, izdela zemljevid in krmari v neurejenem okolju. Glavni cilj je zgraditi 3D-zemljevid ter varno načrtovati pot v neurejenem okolju. Prepričani smo, da je avtomatiziran BLV lahko v večjo pomoč pri iskanju žrtev v primeru potresa v primerjavi z mobilno robotsko bazo, ki ima omejeno gibanje v neurejenih okoljih. Majhen in okreten BLV, se lahko premika po zraku, ima posledično večjo mobilnost kot kopenski mobilni robot. To mu omogoča, da lahko hitreje razišče večje področje. Omenjen sistem je predstavljen v [59], delovanje algoritma pa je mogoče videti na posnetku [60].

BLV v osnovi sestavljajo vetrnice katere skupaj s elektromotorji skrbijo za vzgon mobilnega robota. V našem primeru smo uporabljali kvadrokopter, kar pomeni da ima štiri vetrnice, lahko pa imajo tudi 6 ali 8 vetrnic. Različno število vetrnic vpliva na mobilnost in odzivnost BLV. Zelo pomembna komponenta BLV je seveda baterija, od katere je odvisen čas delovanja – letenja BLV. V tem trenutku manjša BLV lahko dosegajo od 10 do 20 minut letenja. Naslednja komponenta je vgrajen računalnik, kateri skrbi za hitrostno regulacijo vetrnic, ta računalnik sprejema višje nivojske hitrostne ukaze katere potem s pomočjo P, PI ali PID regulatorja izvede na pogonskih elektromotorjih. Za percepcijo okolice BLV uporabljajo kamere, ultrazvočne senzorje, merilce traka, IMU in LMR. S kombinacijo omenjenih senzorjev se BLV lahko znajde v prostorju in tako s pomočjo procesorske enote izvede določeno nalogo samostojno.

Pripravili smo modularno zasnovo navigacijskega in simulacijskega sistema za BLV, kar pomeni, da smo zgradili simulacijo, preiskovanje, načrtovanje poti in sledenje poti. Sistem je modularen, kar pomeni, da je mogoče posamezne sklope ponovno uporabiti ali zamenjati. Slika 54 prikazuje grafični koncept celotnega navigacijskega sistema. Prvo verzijo sistema smo zgradili v simulatorju Gazebo. Za simulacijo BLV smo uporabili simulacijski paket hector_quadrotor [61] v knjižnici ROS. Simulacija Hector_quadrotor vsebuje simulacijo kvadrokopterja in simulirano gravitacijo ter kartiranje Hector z 2D LMR. Kartiranje v 2D za naše potrebe ni bilo primerno, zato smo simulacijski sistem nadgradili s 3D-kartiranjem [62].



Slika 55: Pregled navigacijskega sistema za leteče robote.

Pri našem sistemu smo uporabili dve kameri in majhen, simuliran 3D-senzor VELODYNE VPL16 LITE, ki smo ga simulirali in uporabili v Gazebo. Naslednji korak je bila izdelava modela okolja, tega smo zgradili s pomočjo knjižnice OctoMap, ki je integrirana v ROS. OctoMap izvaja: 3D-pristop kartiranja zasedenosti omrežja, zagotavlja podatkovne strukture in algoritme kartiranja. Ta uporablja oblak točk, ki jih poda 3D LMR v obliki prostostnih objektov, na zelo učinkovit način. Octomap omogoča spreminjanje preproste ločljivosti, optično branje in filtriranje tal, ter omejitev višine glede na začetno točko, kjer se ustvari

osnovni okvir. V tem sistemu je potreben dodaten sistem za lokalizacijo, ker OctoMapping ne vsebuje postopka SLAM, zato smo uporabili zunanji lokalizacijski sistem.

Za namen preiskovanja smo razvili algoritem, ki določa nove raziskovalne točke v 3D-prostoru. Za načrtovanje varne poti do preiskovalnega cilja smo zgradili algoritem za načrtovanje poti. Algoritem za načrtovanje poti izdela prosto pot glede na trenutno stanje zemljevida, ni pa nujno, da ostane enak med gibanjem po prostoru. Zadnji algoritem skrbi za sledenje po dani poti z BLV, sledenje vključuje reaktivno obnašanje, z namenom da se preprečijo nepredvideni trki. V tem primeru se kartiranje izvaja z algoritmom Octomapping, ki za pravilno delovanje potrebuje dodatni sistem za oceno lokacije BLV v prostoru. Čim natančnejša je ocena lokacije, boljše je izdelan zemljevid. Za oceno pozicije uporabljamo OptiTrack sistem, ki ocenjuje položaj robota v prostoru s pomočjo strojnega vida [63].

5.1 PREGLED PODROČJA

Avtorji v [64] predstavljajo navigacijski sistem z oceno položaja, ob tem uporabijo simulacijsko in realno okolje. Bistvenega pomena je, da GPS sistem ni na voljo, ker eksperimente opravljajo v gostih gozdnih okoljih. Za izdelavo zemljevida avtorji uporabljajo 2D LMR in postopek GraphSLAM. V tem primeru imajo pomanjkanje informacij v tretji dimenziji, ker z 2D LMR ne morejo pokriti celotne površine v treh dimenzijah. Sicer dokažejo delovanje navigacijskega algoritma v realnem in simulacijskem eksperimentu. Avtorji v [65] podobno kot v [64] predstavijo delujoč sistem za oceno lokacije BLV v realnem neurejenem okolju, pri tem uporabljajo algoritem scan-matching ali SLAM za oceno pozicije in orientacije BLV. Implementacija s pravim robotom ni bila izvedena, njihova ideja pa sloni na 2D LMR.

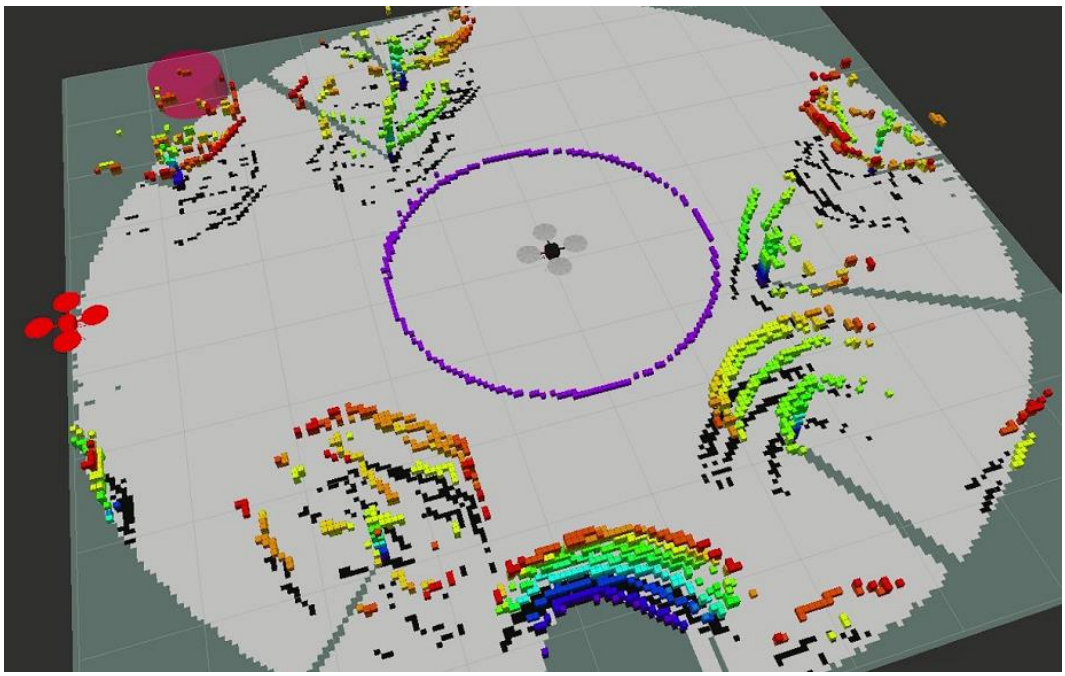
Kinematični model mobilnega robota smo v tem primeru primerjali s kinematičnim modelom kopenskega mobilnega robota, saj so krmilne velikosti podobne (translacijska in kotna hitrost), pri čemer leteči mobilni roboti uporabljajo še eno dodatno translacijsko hitrost, ki neposredno vpliva na višino robota. V večini primerov je podrejena regulacijski zanki hitrosti, izvedene s PID regulatorjem, vendar se to podrobnost pri različnih letečih mobilnih bazah razlikuje.

5.2 PREISKOVANJE

Preiskovanje je ena od osnovnih nalog avtonomnih mobilnih robotov. Osnove za preiskovanje temeljijo na [66–68]. Naloga robota je preiskati neurejeno okolico na hiter in učinkovit način. Hitrost je pomembna

predvsem zaradi omejene avtonomije kvadrokopterja. Za dosego tega cilja smo uporabili preprosto tehniko (angl. *frontear-based*), podobno kot v primeru 2D-modelu okolice.

Območje preiskovanja je omejeno na velikost zemljevida. Od trenutnega položaja robota se širijo žarki, ki streljajo v ravni liniji, dokler ne udarijo v mejo trenutno znanega sveta. Iskanje meja poteka na 360°; na podlagi danega parametra, ki določa gostoto žarkov. Dodaten parameter določa gostoto trčenja glede na velikost robota. Vsak žarek ponudi največ enega potencialnega kandidata za preiskovalni cilj, ki je shranjen v seznamu. Prosto ciljno mesto predstavlja prostor, kjer se robot lahko nahaja brez trčenja. Ta cilj se nahaja v bližini meje znanega/neznanega prostora in izpolnjuje pogoj najbolj oddaljene točke.

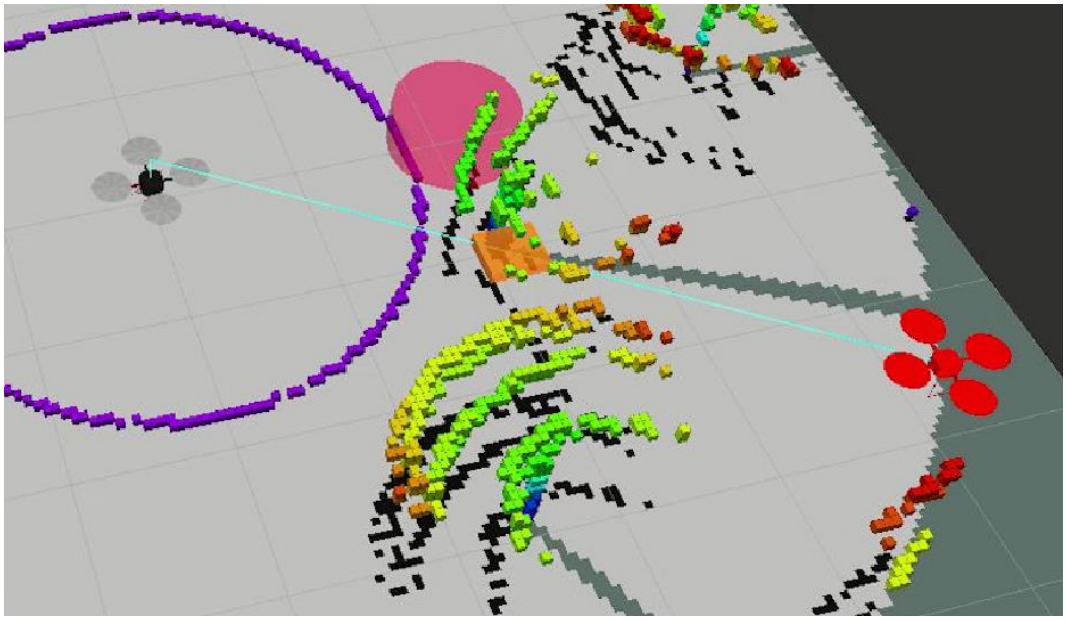


Slika 56: Vizualizacija raziskovalne procedure.

Za določanje ciljne preiskovalne točke uporabimo filter na celem seznamu, na katerem se išče najbližji kandidat za novi preiskovalni cilj. Ohranimo seznam drugih potencialnih ciljev, v primeru, da trenutni kandidat ne more biti dosežen. Slika 56 prikazuje primer raziskovanja v 3D-prostoru. Rdeči model BLV predstavlja ciljno raziskovalno točko, ki je bila določena samodejno. Prozoren rdeči valj predstavlja enega od možnih kandidatov preiskovanja. S tem pristopom smo uspeli preiskati celotno področje v eksperimentalnem in realnem eksperimentu. Preiskovanje se lahko prilagodi s pomočjo parametrov, kar pomeni hitrejši izračun preiskovalnega cilja ali učinkovitejša izbira cilja z daljšim časom preiskovanja.

5.3 NAČRTOVANJE POTI

Ideja za preprosto globalno načrtovanje poti izhaja iz [69]. Navdih za izdelavo našega načrtovanja poti izhaja iz pristopov v [70] in [71]. Preiskovanje nam v prvi fazi poda prost preiskovalni cilj. Nato želimo izračunati prosto pot do določenega preiskovalnega cilja v prostoru. Rezultati izračunov bodo v obliki točk, ki skupaj tvorijo pot. Najprej preizkusimo najkrajšo pot do preiskovalnega cilj, torej ravno pot. Nato preverimo možnost trkov na trenutni poti. Preverjanje trkov poteka podobno kot preverjanje prostega prostora v preiskovalnem delu. Pregledi trčenja se lahko prilagodijo glede na velikost robota in parameter, ki poda število preverjanja trkov na velikost robota. Če se zazna trk, se označi mesto prvega trčenja in nato zadnji zaznani trk, kot je prikazano na sliki 57, kjer je območje trčenja označeno z oranžno barvo. Ko se odkrije območje trčenja, skušamo najti prosto pot okoli njega. To izvajamo z enakimi preverjanji trkov kot v primeru preiskovanja, ki temelji na določenem prostorskem obsegu okoli robota (transparentni rdeči cilinder). Iskanje prostega prostora se odvija okoli oranžnega kvadrata, preverjanje se izvaja pravokotno na najdaljšo stranico kvadrata. V neposredni okolici, ki je določena s parametrom oddaljenosti, se nato preverjajo trki, ki se zapišejo v seznam.



Slika 57: Primer iskanje proste poti, kjer oranžna kocka prikazuje potencialne trke na dani poti.

Sedaj imamo seznam potencialnih prostih mest v okolici oranžnega kvadrata v prostoru, nadalje moramo izbrati, katera pozicija je najbolj primerna za prosto pot mimo ovire. To naredimo tako, da določimo, na kateri strani je, levo ali desno od centra pravokotnika, največ prostih mest. Nato izmed izbranih pozicij izberemo srednjo prosto pozicijo, ki jo definiramo kot vmesno točko na načrtani poti. Ta postopek se

ponavlja, dokler ne najdemo celotne poti brez trčenj do danega cilja. Po končanem prvem krogu pregledov načrtovalec ponovno preveriti celotno pot, da se prepriča, ali res ne more priti do trkov. Če proste poti ni mogoče najti, zaprosi za novi preiskovalni cilj.

5.4 SLEDENJE POTI

Kot pri načrtovanju poti, smo tudi za sledenje poti preučili [71]. Preprosto sledenje poti smo izvedli na podlagi [72] in [73]. Pomemben del sledenja poti predstavlja reaktivno obnašanje, ki skrbi, da robot ne zadane ovir, ki so bile pri načrtovanju skrite.

Reaktivno obnašanje

Z reaktivnim vedenjem se najde varna pot brez trčenj, tudi če so nove ovire dinamične. V nasprotnem primeru izvedba poti ni uspešna in sledenje poti sporoči, da ni mogoče doseči cilja. Reaktivni del sledenja poti poteka na osnovi valja in je definiran s parametrom obsega valja. Od trenutnega položaja robota se generirajo žarki, ki preverjajo zasedenost celic v Octomapu. Trenutno se reaktivne sile izračunavajo petkrat na sekundo. Reaktivno obnašanje želi slediti dani poti, vendar ovire na dani poti delujejo odbijajoče, kar imenujemo reakcijske sile, Xf, Yf, Zf in Df se izračunajo v vsaki dimenziji, kot je prikazano v (49).

$$Xf = avg(Xv)$$

$$Yf = avg(Yv)$$

$$Zf = avg(Zv)$$

$$Df = avg(Dv) \tag{49}$$

Vrednosti Xv, Yv in Zv predstavljajo razdaljo od centra robota do zasedene celice, Dv predstavlja povprečno razdaljo do zasedenih celic.

Izvedba poti s pomočjo ukazov hitrosti

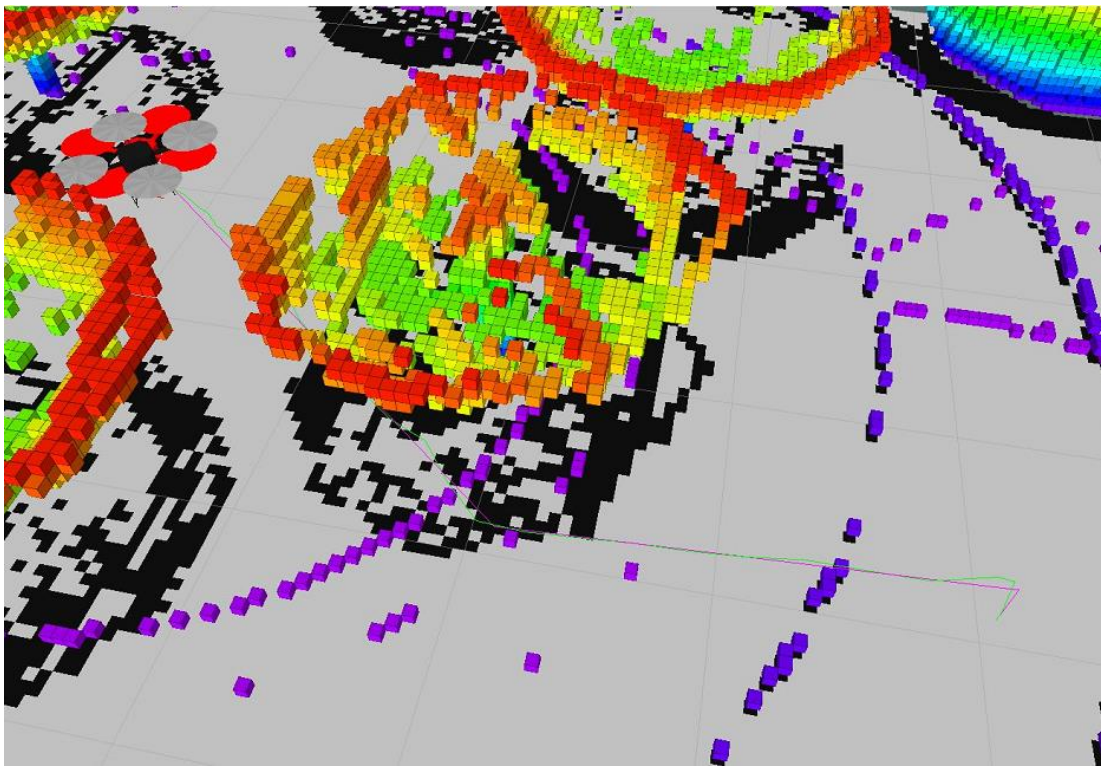
Izračun hitrosti mora biti izveden, kakor hitro je mogoče. Zato nudimo posodobitev izračuna, ki temelji na posodobitvi položaja – ko je na voljo podatek o položaju, se izračunajo nove hitrosti. Ukaz hitrosti se po navadi izračuna na podlagi naslednje lokacije točke poti. Naslednja točka poti je izračunana na podlagi najbližje točke poti do trenutnega položaja robota. To pomeni, da v vsaki ponovitvi najdemo najbližjo razpoložljivo točko poti in vzamemo naslednjo v vrsti, ki predstavlja nov cilj za robota. Ukaz hitrosti se

generira na osnovi (50). Parameter $react$ opredeljuje reaktivni vpliv sile na obnašanje robota, ki je odvisen od $lookahead$, Df in $robcol_x$. Parameter $robcol_x$ predstavja premer robota. Enak postopek velja za dimenziji Y in Z.

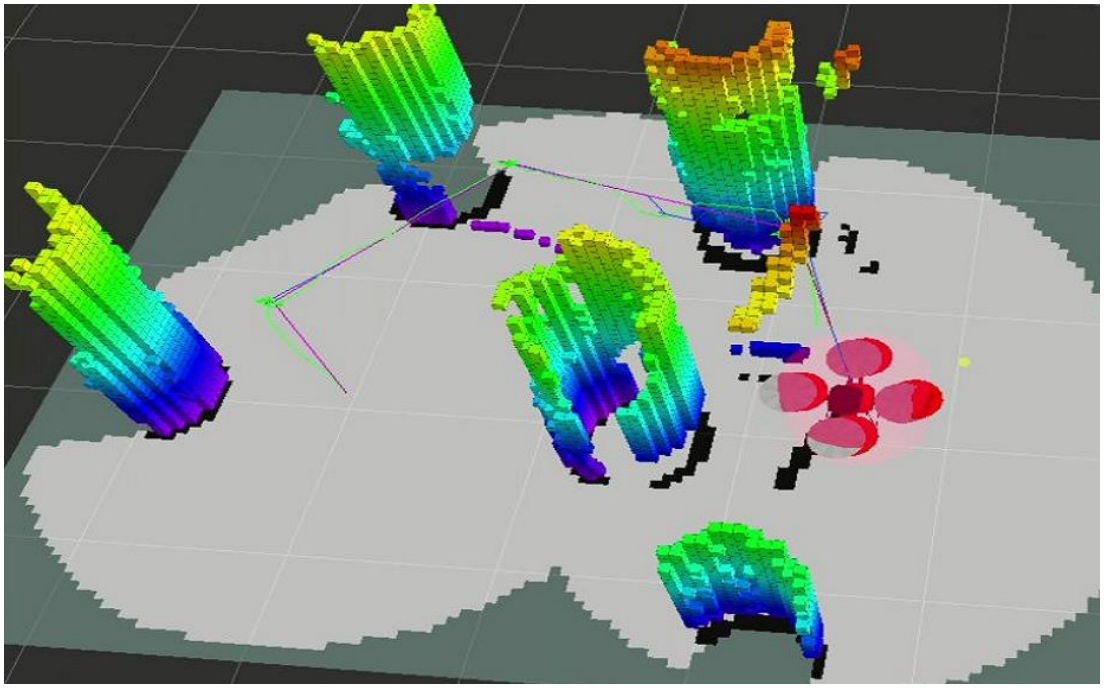
$$react_x = react * (maxlinvel * \frac{Pvn * Df + \frac{robcol_x}{2}}{Pvn} \tag{50}$$

$$cmd_x = goal_p * (maxlinvel * (\cos(an_h) + react_x) \tag{51}$$

Parameter Df se upošteva, če je ovira prisotna v bližni robota. V tem primeru sledenje poti proizvaja odstopanja v obliki hitrosti, da bi se izognili nepredvidljivim oviram, ki se pojavijo na poti. Primer tega obnašanja lahko vidimo na sliki 58 in 59. Vijoličasta pot je načrtovana pot in zelena pot je izvedena pot robota. Reaktivni del $react_x$ se izračuna samo, če je zazna ovira v bližini robota. Reaktivne hitrosti so izračunane glede na največjo linearno hitrost, ki se prilagaja glede na bližino zasedenih celic. Enak postopek izračuna cmd_x velja za hitrost ukaznega izračuna v osi Y cmd_y in Z cmd_z.



Slika 58: Trajektorija robota (zeleno) in načrtovana pot (vijoličasta) med navigacijo v simulacijskem okolju.

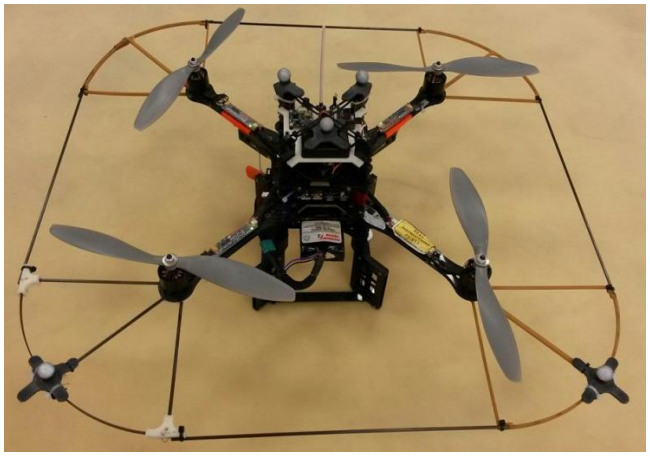


Slika 59: Izvedba avtonomnega preiskovanja v simulacijskem okolju.

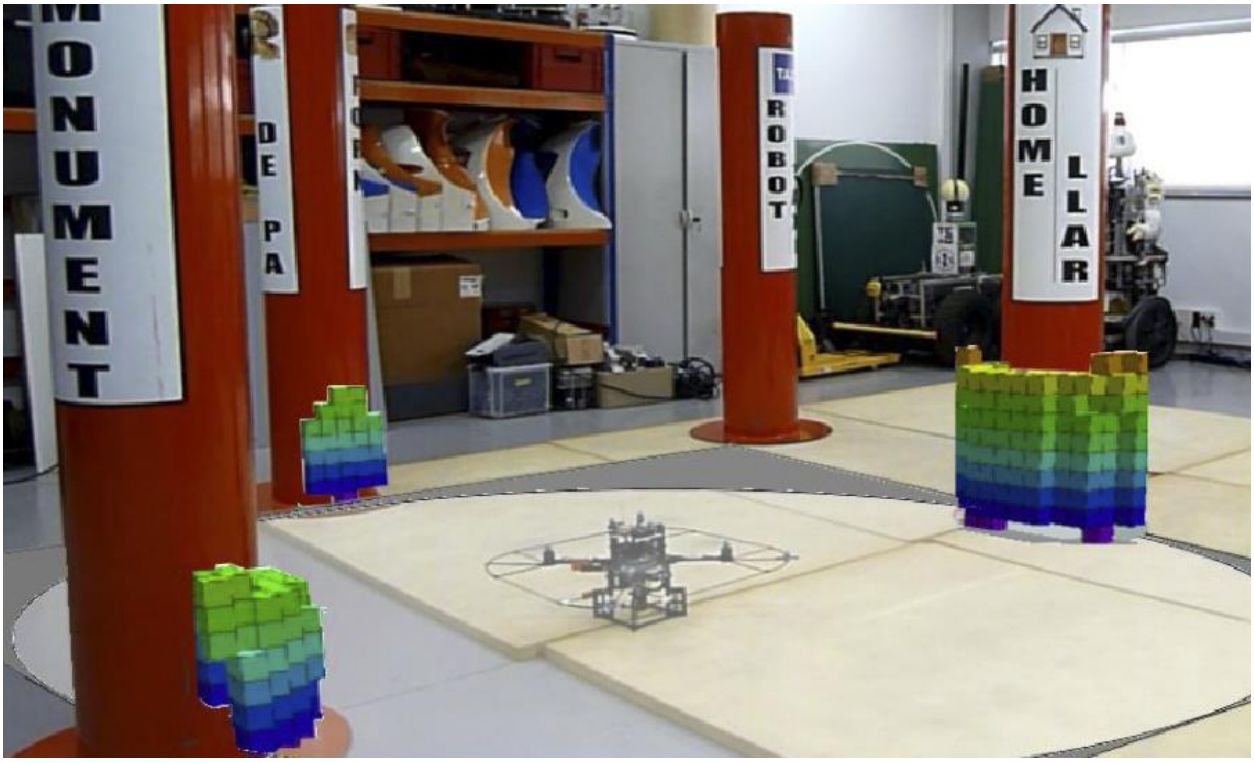
5.5 EKSPERIMENT Z VIRTUALNO RESNIČNOSTJO

Za eksperiment z virtualno resničnostjo smo se odločili predvsem iz varnostnih razlogov, saj je v tem primeru poškodba opreme malo verjetna. Zgradili smo simulacijsko okolje za razvoj in preizkušanje navigacijskih algoritmov. V realnem eksperimentu je bil uporabljen pravi BLV v nadzorovanem laboratorijskem okolju. Vse algoritme za navigacijski sistem in vodenje robota smo zgradili v ROS-u. Za realni eksperiment smo uporabili pravega robota na osnovi [74], imenovanega Kinton, prikazan na sliki 60. Robotska baza lahko doseže maksimalno hitrost 16 m/s in največjo hitrost vzpenjanja 8 m/s. Za komunikacijo je bila uporabljena 2,4 GHz povezava XBee, integriran procesor Intel Core 2. Robot Kinton je opremljen s kamero in z virtualnim 3D LIDAR senzorjem Velodyne VLP-16. Robotska baza meri 651 mm x 651 mm x 188 mm in ima 650 g maksimalne nosilnosti. Realno okolje nastavitev z robotom Kinton je prikazano na sliki 61. Slednja prikazuje realno okolje z BLV v kombinaciji z virtualno resničnostjo. To okolico smo rekonstruirali zato, da realne dimenzije okolja uvozimo v simulacijsko okolje Gazebo. Simulirani robot je bil povezan z realnim robotom preko pozicije, ki jo je priskrbel natančen merilni sistem Optitrack. Navigacijski sistem temelji na modelu virtualnega okolja, zgrajen z 3D-octomap, v katerem lahko robot izvaja preiskovanje, načrtovanje poti in izvedbo poti v realnem okolju. Slika 62 – levo zgoraj prikazuje rekonstruiran laboratorijski model okolice, ki je bil uporabljen v virtualni resničnosti. V ta model okolja

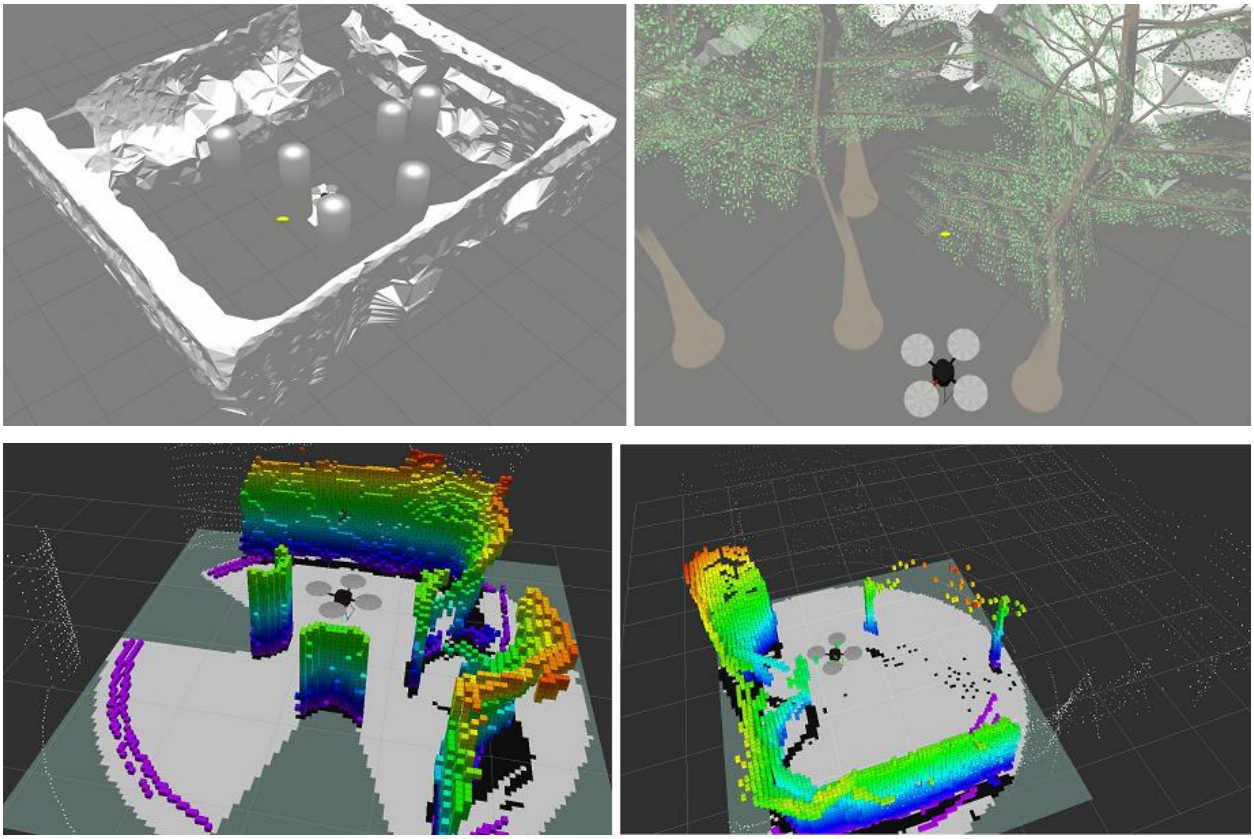
smo dodali preproste stebre ali kompleksne 3D-modele, kot lahko vidimo na sliki 62 – zgoraj. Model okolja je bil izdelan z octomap_server na podlagi virtualnega 3D LMR, ki je na voljo kot vtičnik za Gazebo. Slika 62 – spodaj prikazuje primer preprostih stebrov, ki se zgradijo med gibanjem robota, enako velja za kompleksno strukturo dreves.



Slika 60: Robot Kinton, na osnovi AscTec Pelican BLV.

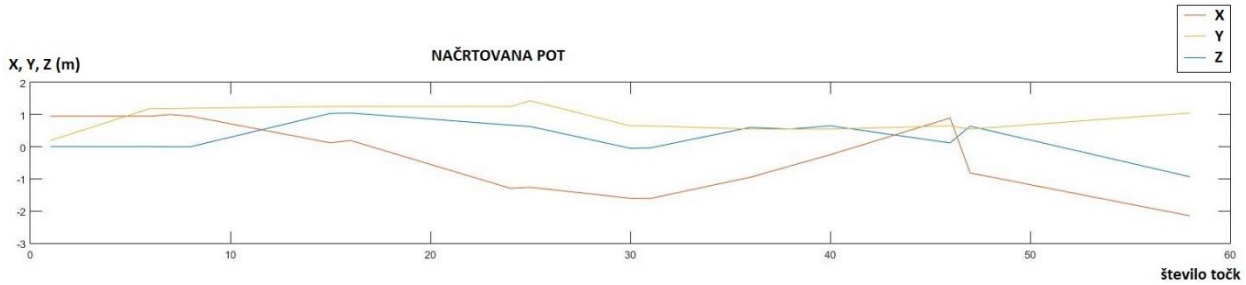


Slika 61: Primer kombinacije virtualnega in realnega okolja z vključenimi stebri in mobilnim robotom.

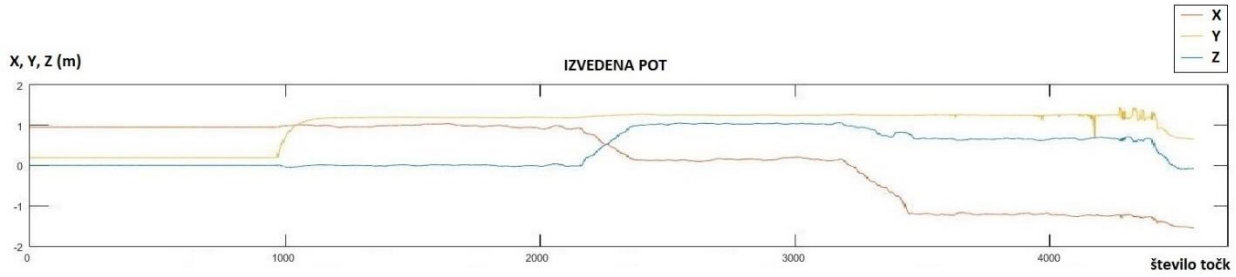


Slika 62: Levo navigacija med enostavnimi virtualnimi stebri, desno navigacija med kompleksnimi simulacijskimi drevesi.

Za oceno in analizo vedenja sistema v realnem svetu smo posneli vse realne pozicije robota in točke načrtovane poti, kot je prikazano na sliki 63 in sliki 64. Graf v rdeči barvi prikazuje os X, oranžni os Y in modri os Z. Opazimo lahko, da robot ni dosegel načrtovane točke in prav tako je med potovanjem odstopal od načrtovane poti. To je bilo predvsem zaradi aktivnega reakcijskega obnašanja robota. Leteči mobilni robot je zaradi bližin ovir burno odzival na okolico in s tem prilagajal svojo trajektorijo.



Slika 63: X-rdeča, Y-oranžna, Z-modra, prikazana načrtovana pot v odvisnosti od števila podanih XYZ točk.



Slika 64: X-rdeča, Y-oranžna, Z-modra: prikazana izvedena pot v odvisnosti od števila podanih XYZ točk.

V tem delu smo predstavil samostojen navigacijski sistem za leteče robote, s katerim lahko varno navigira v neurejenih okoljih. Celotni sistem vključuje znane, preizkušene algoritme, ki smo jih prilagodili za navigacijo v 3D-okolju. V razvojno delo smo dodatno vključili t. i. virtualno resničnost, s katero smo simulirali okolje in v njej varno preizkušali navigacijske algoritme za leteče mobilne robote. To pomeni, da lahko delovanje mobilnega robota BLV enostavno preizkušamo ob virtualnih ovirah kakršne koli oblike. V tem primeru ni nevarnosti, da bi robot ob napaki navigacijskega sistema trčil v oviro in se poškodoval.

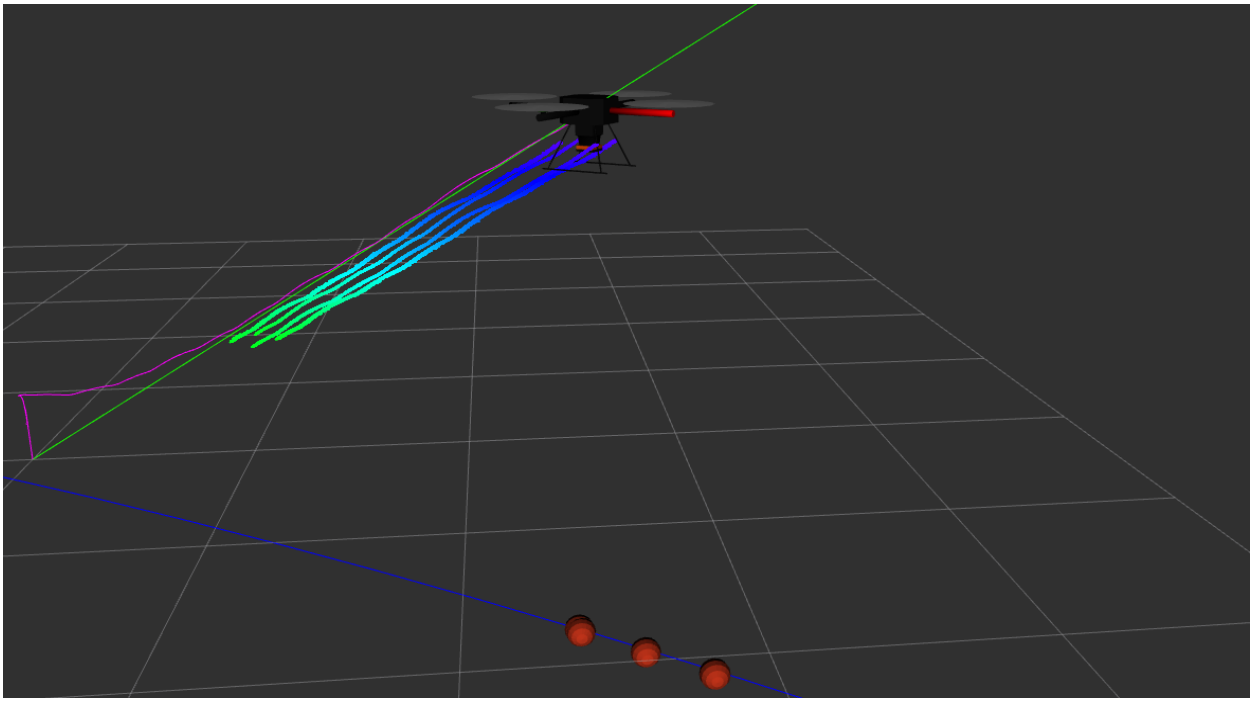
5.6 SIMULACIJSKI EKSPERIMENT

Delo smo nadaljevali s predelavo algoritma DPKL za leteče robote. S tem smo želeli dokazati široko uporabnost pristopa DPKL za sledenje poti. Zgradili smo virtualno začrtano pot, ki smo ji želeli slediti z BLV. Uporabili smo enak postopek, kot je opisan v poglavju 4, za ravnino X-Y. V tretjo dimenzijo smo implementirali izračune razlike odstopanja od želene referenčne višine in na podlagi teh podatkov dodali hitrost motorjem BLV. Kvadrokopter smo vozili v X-Y ravnini z izračunom hitrosti enako kot pri 2D-sistemih, za izračun linearne hitrosti v osi Z pa smo uporabili enačbo (52).

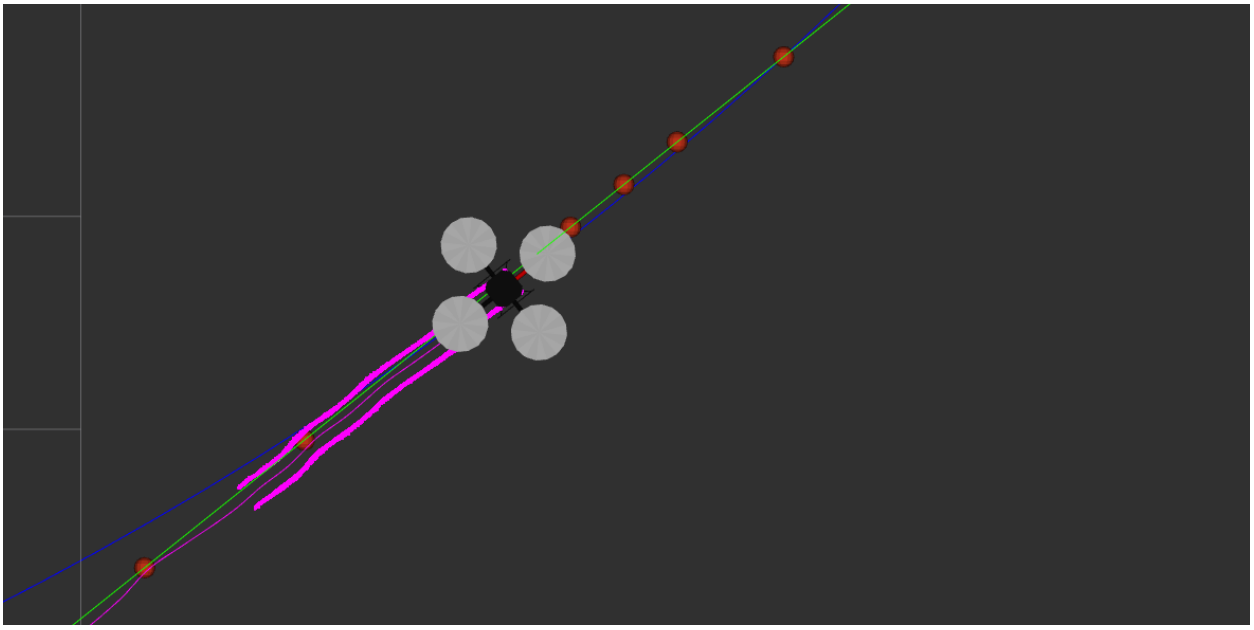
$$cmd_z = diff_z * inc_{vel} \tag{52}$$

$$diff_z = next_z - curr\_z \tag{53}$$

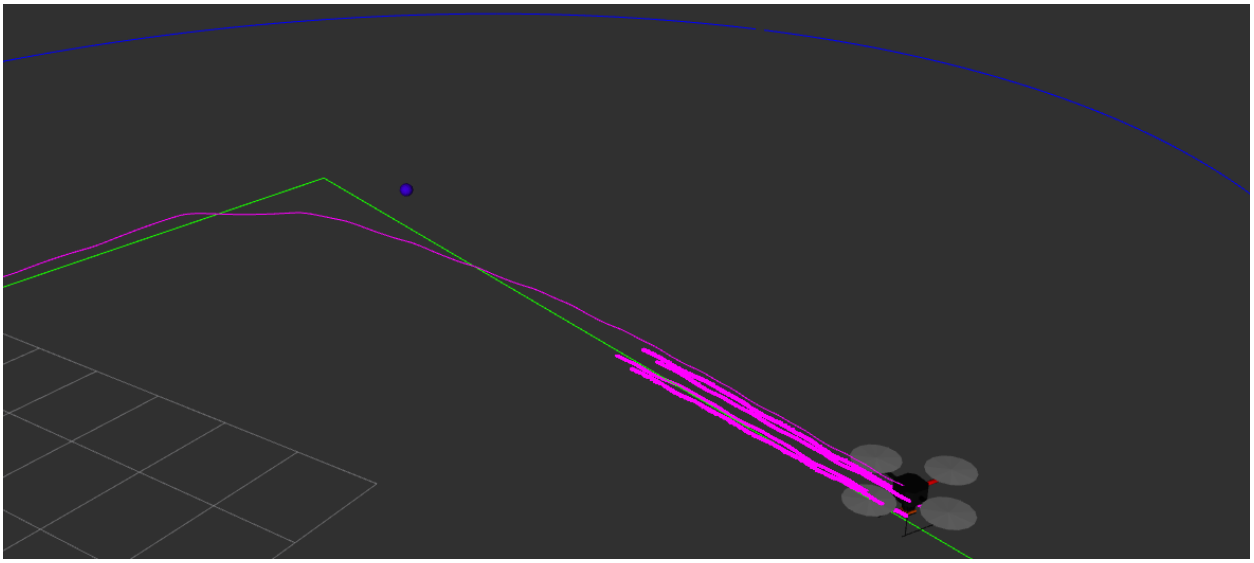
Parameter $diff_z$ predstavlja razliko med trenutno lokacijo mobilnega robota in naslednjo najbližjo točko, in to samo v dimenziji Z. Slike 65 – 67 prikazujejo simulacijsko okolje, zelena črta prikazuje želeno pot, ki ji mora BLV slediti. Moder krog predstavlja lok, ki je trenutno izračunan s polmerom R. Vijoličasta črta predstavlja trajektorijo mobilnega robota. Rjave krogle predstavljajo točke na dani trajektoriji in so projicirane na ravnino X-Y z namenom, da lahko izračunamo radij R za sledenje. Na sliki 65 vidimo trajektorijo z začetnim dvigom BLV; dvig je potreben za varno in normalno lebdenje v zraku.



Slika 65: Prikaz letečega robota ob pričetku sledenja poti (zelena), vijoličasta črta predstavlja trajektorijo letečega mobilnega robota, modra črta pa predstavlja projekcijo izračunane krožnice, kateri robot sledi.

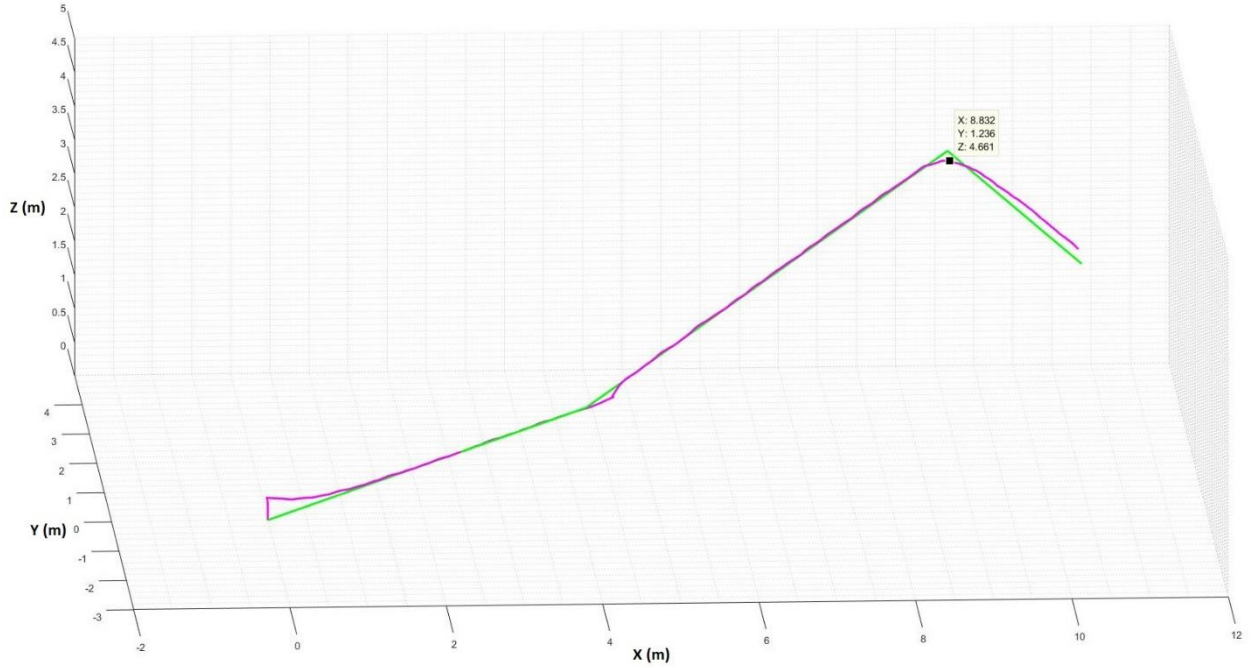


Slika 66: Prikaz sledenja letečega mobilnega robota s ptičje perspektive.

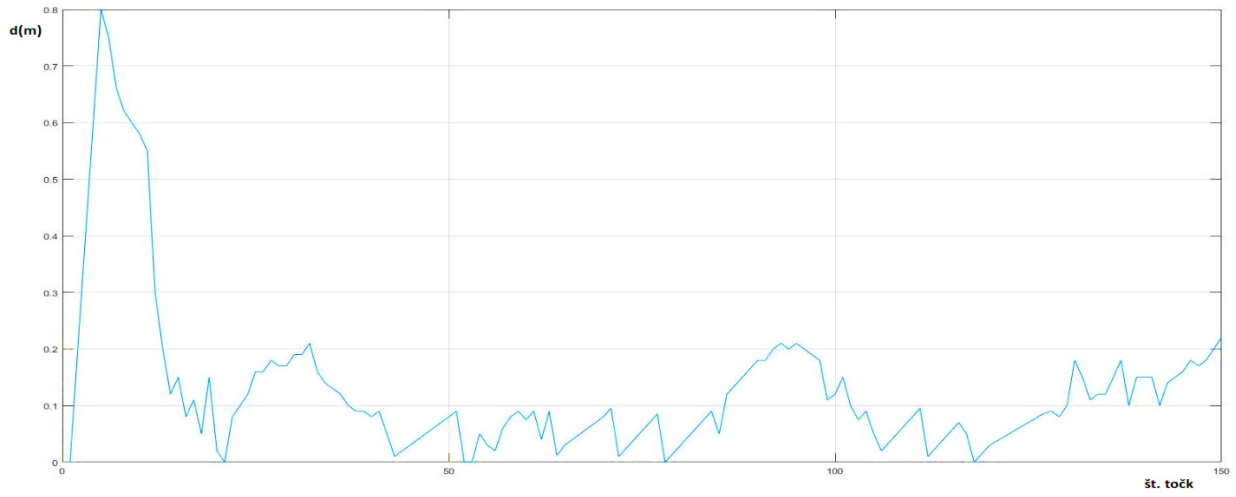


Slika 67: Prikaz sledenja letečega mobilnega robota med prehodom iz dviga v spust pri sledenju poti.

Postopek sledenja v 3D-prostoru je preslikan v 2D-prostor. V 2D-prostoru je postopek sledenja poti popolnoma enak, kot je opisan v poglavju 4. Algoritem za sledenje smo preizkusili z načrtano potjo. Rezultati sledenja v simuliranem okolju v 3D so prikazani spodaj.



Slika 68: Grafični prikaz odstopanj od sledenja poti (vijoličasta) glede na načrtano pot (zelena).



Slika 69: Prikaz absolutnih razlik med trajektorijo robota in načrtano potjo.

Graf na sliki 68 z zeleno barvno prikazuje načrtano pot v 3D-prostoru in z vijoličasto označeno trajektorijo, ki jo je simulirani robot izdelal med sledenjem. Pri vzletu opazimo večje odstopanje, to je zaradi varnega vzleta BLV. Vzleteti mora na določeno višino in nato prične s sledenjem načrtani poti. Med sledenjem poti smo opazili največje absolutno odstopanje od načrtane poti 0,2101 m. Odstopanje je precej majhno v primerjavi z rezultatom v prejšnjem poglavju, ko smo seveda uporabili pravega robota. V realnih pogojih je zelo težko izvajati zelo natančno regulacijo hitrosti in sledenje načrtani poti. Povprečno odstopanje od dane poti je bilo 0,128 m. S tem pristopom se želimo približati čim bolj natančnemu in odzivnemu sledenju letečih robotov. Omenjeni primer je mogoče videti v posnetku [75].

Hipoteza **H4** predpostavlja letečega mobilnega robota, ki je zmožen samostojno izvesti raziskovanje prostora, kar smo uspeli doseči. Uspešno smo integrirali obstoječe algoritme za izdelavo zemljevida, načrtovanje poti in izvedbo poti s upoštevanjem dinamičnih ovir. Hipoteza **H4** predvideva, da je lahko DPKL algoritem, ki je bil v osnovi bil razvit za kopenske mobilne robote, uporabljen tudi za leteče mobilne robote. V primeru sledenja poti, kjer ni bilo prisotnega reakcijskega obnašanja smo dosegli povprečno odstopanje 0.128m od želene trajektoriji. Odstopanje od želene trajektorije je za letečega mobilnega robota zadovoljivo. Na podlagi predstavljenih rezultatov smo DPKL algoritem vključili v avtonomni sistem letečega mobilnega robota, zato lahko potrdimo hipotezo **H4**.

6 ZAKLJUČEK

V doktorski nalogi je predstavljen razvoj reševalnega mobilnega robota in njegovi algoritmi za delovanje v neurejenih okoljih. Mobilni robot je sestavljen iz več sistemov, ki so bistveni za njegovo avtonomno delovanje: laserski merilnik razdalje, senzorska glava in diagnostična plošča. S pomočjo teh sistemov z algoritmi izvaja gibanje v neurejenem, neznanem in nepredvidljivem okolju in hkrati uspešno odkriva žrtve na podlagi izdelanega zemljevida s pomočjo postopkov LMR in SLAM. Podrobneje je predstavljen algoritem za zanesljivo sledenje poti za mobilne robote z diferencialnim pogonom. Opisani so tudi sistemi za uravnavanje laserskega merilnika razdalje ter sistem senzorske glave in diagnostične plošče, ki smo jih razvili v okviru našega dela. Delo je bilo usmerjeno v načrtovanje poti v neurejenih okoljih, kjer smo delo nadaljevali z letečimi mobilnimi roboti. Tukaj smo implementirali podobne rešitve v 3D-prostoru.

Krožno gibanje je naravno gibanje mobilnega robota, na tej osnovi lahko pričakujemo učinkovito rešitev težav s sledenjem podani poti. Predpostavili smo, da je mogoče na osnovi krožnega gibanja zasnovati izvedbeno dovolj enostaven algoritem. Algoritem naj bi omogočal hitro izvedbo in odzivnost robotske baze na odstopanja vožnje od začrtane poti. Algoritem za sledenje poti smo zasnovali na osnovi krožnega gibanja v 2D- in 3D-prostoru. Algoritem deluje z visoko frekvenco iskanja ujemanja krožnic z danimi točkami v prostoru, ki predstavljajo načrtano pot v prostoru. Algoritem, zasnovan osnovi hipotez, smo preverjali na svetovnem tekmovanju RoboCup 2014 v Braziliji. Omenjeni algoritem je omogočil dokaj velikemu goseničnemu mobilnemu robotu, da je avtonomno premagoval kompleksne ovire v reševalni areni. To se je dokazalo z največjim avtonomno izdelanim zemljevidom na tekmovanju.

Z danim algoritmom želimo dokazati zanesljivost in natančnost sledenja tudi v neurejeni okolici.

S sistemom za uravnavanje laserskega merilnika želimo omogočiti kopenskemu mobilnemu robotu, da premaguje ovire in stopnice vse do +/-45° v X in Y koordinatnih oseh robota in pri tem izdela natančen načrt brez napak. Omenjeni sistem uravna laserski merilnik do te mere, da je vedno v vodoravnem položaju.

S sistemom senzorske glave želimo omogočiti večjo svobodo pri iskalnih strategijah oziroma pri pokrivanju okolice robota. S tem bi povečali verjetnost, da se najde žrtev. Tako želimo implementirati napredni algoritem, ki je na podlagi dinamike izdelanega zemljevida prilagodil iskalno strategijo. Ta je predvidevala, da so žrtve v večini primerov zagozdene v stenah. Torej smo se osredotočili na trenutno stanje okolice robota in temu prilagodili iskalno strategijo.

Diagnostična plošča nadzoruje sisteme robota med delovanjem. V reševalni areni se pogosto zgodi, da robot trči in se servomotorji za prostostne stopnje preobremenijo, to želimo avtomatsko popraviti s pomočjo diagnostične plošče, ki ponastavi servomotor in tako omogoči nadaljevanje naloge brez posega operaterja. Delovanje diagnostične plošče smo preverili z eksperimenti.

Za navigacijo BLV je potreben navigacijski sistem z algoritmi , ki zmorejo varno in hitro manipulirati z danim BLV. Navigacijski sistem sestoji iz raziskovanja, načrtovanja in izvedbe poti z letečim mobilnim robotom. Navigacijski sistem je bil preizkušen v simulaciji in v virtualnem okolju, kjer je bil vključen realni leteči robot.

6.1 POTENCIALNA UPORABNOST

Algoritem za sledenje s pomčjo DPKL lahko uporabimo na kateri koli mobilni bazi z diferencialnim pogonom (gosenični, kolesni, leteči). Avtonomni navigacijski sistemi se lahko uporabljajo za različne naloge, npr. reševanje po potresu, nadzorni mobilni sistem, avtonomni avtomobili in podobno. Navigacijski sistem za BLV je potencialno uporaben v nadzornih nalogah: varovanje mej, iskanje ponesrečencev, prinašanje zalog na težko dostopne prostore, raznašanje pošte in morda sestavljanje visoko ležečih konstrukcij. V zadnjih letih so se v industrijskih okoljih pojavili avtonomni mobilni sistem, ki uporabljajo open-source sisteme, na katerih smo gradili vsa ta leta. Ti industrijski mobilni roboti so v pomoč pri logistiki v podjetju ali celo manjkajoči del popolnoma avtonomne proizvodnje.

6.2 PREDLOGI ZA NADALJNO DELO

Področje navigacijskih sistemov za avtonomne mobilne robote je zelo široko. Velik del področja smo obdelali v tej disertaciji, še vedno pa vidimo velike možnosti za raziskave na področju sledenja poti za različne baze ter na področju diagnostične plošče, ker je velika verjetnost, da bo nekoč diagnostična plošča sestavni del vsakega robota. Navigacijski algoritmi za leteče robote so še dokaj mlado področje, ki se lahko razširi predvsem na sledenje poti ter optimiziranje in dograjevanje algoritmov za avtonomno navigacijo v neurejenih okoljih.

6.3 DOPRINOS K ZNANOSTI

Predstavljeno delo obravnava zelo široko področje mobilne robotike. Treba je prepoznati in obvladovati različna področja znanosti, kot so mehanika, elektrotehnika, elektronika in računalništvo. Doprinos vidimo predvsem na področjih razvoja sistema za aktivno diagnostiko in samopopravljivih scenarijih. Predpostavljamo, da bodo nekoč avtonomni sistemi morali vsebovati podobne elektromehanske elemente, ki bodo zmožni nadzorovati pravilnost delovanja posameznih sistemov.

Naslednji večji del, ki ga izpostavljamo, je sledenje poti z kopenskim in letečim mobilnim robotom, kjer uporabljamo pristop DPKL . Algoritem DPKL je smiseln in enostaven za implementacijo. Predstavlja zelo pomemben del pri avtonomni navigaciji robotov, ki so lahko širše uporabljeni.

7 LITERATURA

[1] S. Uran, G. Steinbauer, P. Lepej, J. Maurer: Roboti za reševanje po potresu, Zbornik 15. mednarodne multikonference Informacijska družba - IS 2012, 8.–12. oktober 2012, Ljubljana, Slovenia: zvezek A, (Informacijska družba, ISSN 1581-9973). Ljubljana: Institut Jožef Stefan, 2012, str. 333–336. [COBISS.SI-ID 16349974]

[2] J. Maurer, G. Steinbauer: Autonomous risk-aware exploration, Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on., 21–26 Oct. 2013.

[3] R. Siegwart, I. R. Nourbakhsh: Introduction to Autonomous Mobile Robots, Massachusetts Institute of Technology, A Bradford Book, The MIT Press, 2004.

[4] J. L. Martínez, A. Mandow, J. Morales, S. Pedraza, A. García-Cerezo: Approximating Kinematics for Tracked Mobile Robots, The International Journal of Robotics Research, Vol. 24, No. 10, October 2005, pp. 867–878.

[5] P. Lepej, J. Maurer, G. Steinbauer, S. Uran: Vožnja mobilnega robota z gosenicami po razgibani podlagi in stopnicah = Driving tracked mobile robot over complex enviromet and stairs. V: ZAJC, Baldomir (ur.), TROST, Andrej (ur.). Zbornik dvaindvajsete mednarodne Elektrotehniške in računalniške konference ERK 2013, 16.–18. september 2013, Portorož, Slovenija, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., ISSN 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE, 2013, zv. B, str. 99–102. [COBISS.SI-ID 17175830]

[6] P. Lepej, J. Maurer, G. Steinbauer, S. Uran: Analysis of laser sensor system for a rescue robot. V: ZAJC, Baldomir (ur.), TROST, Andrej (ur.). Zbornik enaindvajsete mednarodne Elektrotehniške in računalniške konference ERK 2012, 17.–19. september 2012, Portorož, Slovenija, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., ISSN 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE, 2012, zv. B, str. 127–130, ilustr. [COBISS.SI-ID 16306966]

[7] http://www.nist.gov/index.html, dostop 15. 10. 2012

[8] C. Castillo, C Chang: A Method to Detect Victims in Search and Rescue Operation using Tempalte Matching, Proceedings of the 2005 IEEE International Workshop on Safety and Rescue Robotics Kobe, pages 201–206, Japan, 2005.

[9] L. Xia, C. C. Chen, J.K. Aggarwal: Human Detection Using Depth Information by Kinect, The Unuversity of Texas at Austin, Department of Electrical and Computer Engineering.

[10] L. Spinello, K. O. Arras: People Detection in RGB-D Data, 2011 IEEE/RSJ International Conference on Inteligent Robots and Systems, pages 3838–3843, September 25–30, San Francisco, CA, USA, 2011.

[11] G. Nejat, Z. Zhang: Finding Disaster Victims: Robot-Assisted 3D Mapping of Urban Search and Rescue Enviroments via Landmark Identification, Departmetnt of Mechatronical Engineering, State Universty of New York at Stony Brook, 1-4244-0342-1/06 2006 IEEE, ICARCV 2006, NY, USA, 11794-2300,2006.

[12] Z. Zhang, H. Guo, G. Nejat, P. Huang: Finding Disaster Victims: A Sensory System for Robot-Assisted 3D Mapping of Urban Search and Rescue Enviroments, 2007 IEEE International Conference on Robotics and Automation, pages 3889–3894, FrB12.4, 10–14 April, Roma, Italy, 2007.

[13] http://ntuzhchen.blogspot.com/2010/12/how-kinect-works-prime-sense.html, dostop 15. 10. 2012

[14] J. V. Hajnal, D. L. G. Hill, D. J. Hawkes: Medical Image Registration, CRC Press LLC, 2001.

[15] P. Lepej, J. Maurer, G. Steinbauer, S. Uran S. Zaman: An integrated diagnosis and repair architecture for ROS-based robot systems, DEARDEN, Richard (ur.), SNOOKE, Neal (ur.). Proceedings of the 23th International Workshop on Principles of Diagnosis (DX-2012): Malvern, UK, July 31 to August 3, 2012. [S. l.: s. n., 2012], str. 35–42. [COBISS.SI-ID 16256278]

[16] S. Zaman, G. Steinbauer , J. Maurer, P. Lepej S. Uran: An integrated model-based diagnosis and repair architecture for ROS-based robot systems, IEEE International Conference on Robotics and Automation, May 6–10, 2013 Karslruhe, Germany. ICRA 2013. [S. l.]: IEEE, cop. 2013, str. 482–489, doi: 10.1109/ICRA.2013.6630618. [COBISS.SI-ID 17645334]

[17] S. Zaman, P. Lepej: MBD counterpart diagnostic board: its hardware, protocol, software, simulator and application, 22nd International Workshop on Robotics in Alpe-Adria-Danube Region [also] RAAD 2013, September 11–13, Portorož, Slovenia. NEMEC, Bojan (ur.), ŽLAJPAH, Leon (ur.). Proceedings. 1st ed. Ljubljana: Jožef Stefan Institute, 2013, str. 86–93. [COBISS.SI-ID 17242390]

[18] S. Zaman, P. Lepej: ROS-based diagnostic board for detecting and repairing hardware faults in autonomous mobile robots, IEEE International Conference on robotics and emerging allied technologies in engineering (iCREATE): proceedings. [S. l.]: IEEE, cop. 2014, str. 95–102, doi: 10.1109/iCREATE.2014.6828347. [COBISS.SI-ID 17978646]

[19] M. Quigle, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, N. Andrew: ROS: an open-source Robot Operating System, ICRA Workshop on Open Source Software, Willow Garage, Menlo Park, California, 2009

[20] G. Grisetti, C. Stachniss, W. Burgard: Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters, IEEE Transactions in Robotics, 2006.

[21] G. Grisetti, C. Stachniss, W. Burgard: Improving Grid-based SLAM with Rao Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling, Univerity of Freiburg, Department of Computer Science.

[22] S. Kohlbrecher, J. Meyer, K. Peterson, T. Graber: Hector SLAM for robust mapping in USAR enviroments, ROS RoboCup Rescue Summer School Graz, 2012.

[23] S. Kohlbrecher, J. Meyer, O. von Stryk, U. Klingauf: A Flexsible and Scalable SLAM System with Full 3D Motion Estimation, Proceedings of the 2011 IEEE International Symposium on Safety, Security and Rescue Robotics, Japan, 2011.

[24] P. Lepej, J. Rakun: Simultaneous localisation and mapping in a complex field environment. Biosystems engineering, ISSN 1537-5110. [Print ed.], October 2016, vol. 150, str. 160–169, ilustr. [COBISS.SI-ID 4202540]

[25] B. Yamauchi: A Frontier-Based Approach for Autonomous Exploration, Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, DC 20375-5337

[26] T. Motaln, Avtonomno raziskovanje mobilnega robota z meta-operacijskim sistemom ROS (Robotski operacijski sistem), Univerza v Mariboru, Fakulteta za strojništvo, 2013.

[27] S. Kohlbrecher, J. Meyer, K. Peterson, T. Graber, U. Klingauf, O. von Stryk: Hector Open Source Modules for Autonomous Mapping and Navigation with Rescue Robots, Department of Computer Science, Department of Mechanical Engineering, TU Darmstadt, Germany, 2014.

[28] S. Kohlbrecher,J. Meyer,U. Klingauf,O. Stryk: A Flexible and Scalable SLAM System with Full 3D Motion Estimation, Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), pp. 155–160, November 1–5, 2011.

[29] ROS hector_navigation stacks [Internet]. [updated on 16 Maj 2014]. Available from: wiki.ros.org/ hector_navigation. Dostop 24. 4. 2017.

[30] H. Choset, K. M. Lynch, S. Hutchinson,G. Kantor, W. Burgard, L. E. KavrakiS. Thrun: Principles of Robot Motion. Cambridge, Massachusetts, A. Bradford Book, The MIT Press. pp 440–472, 2005

[31] F. Lamiraux, J. P. Laumond: Smooth Motion Planning for Car-Like Vehicles, IEEE Transaction on Robotics and Automation, Vol. 17, NO. 4, 2001

[32] J. P. Laumond, P. E. Jacobs, M. Taix, R. M. Murray: A Motion Planner for Nonholonomic Mobile Robots, IEEE Transaction on Robotics and Automation, Vol. 10, NO. 5, 1994

[33] A. Yazici: A Smooth Tour Construction Approach for a Mobile Robot with Kinematic Constrains, Int. j. adv. robot. syst. Vol. 10, 2013

[34] S. Lee, J. P. Hyeon: Dynamic Path-Following Using Temporally Path Generator for Mobile Robots with Nonholonomic Constrains, Mechatronics La., Precision Mechanics Eng., Hanyang Univ., Seoul, South Korea.

[35] F. Diaz del Rio, G. Jimenez, J. L. Sevillano, S Vicente, A. CivitBalcells: A Generalization of Path Following for Mobile Robots, Facultad de Informatica, Univ. Sevilla, Spain.

[36] G. Indiveri, A. Nuchter, K. Lingermann: High Speed Differential Drive Mobile Robot Path Following Control With Bounded Wheel Speeds Commands, IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April, 2007

[37] K. Komoriya K, K. Tanie: Trajectory Design and Control of a Wheel-type Mobile Robot Using B-spline Curve, IEEE International Workshop on Intelligent Robots and Systems, Tsukiba, Japan, September 4–6, 1989

[38] ROS pose_follower software package [Internet]: wiki.ros.org/pose_follower. Dostop 24. 4. 2017.

[40] J. Morales, L. J. Martínez, A. M. Martínez, A. Mandow: Pure-Pursuit Reactive Path Tracking for Nonholonomic Mobile Robots with a 2D Laser Scanner, Hindawi Publishing Corporation EURASIP Journal on Advances in Signal Processing, Volume 2009.

[41] O. Amidi: Integrated Mobile Robot Control, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1990

[42] P. B. Gerkey, K. Konolige: Planning and Control in Unstructured Terrain, Willow Garage, SRI Artificial Intelligence Centre.

[43] D. Fox, W. Burgard, S. Thrun: The Dynamic Window Approach to Collision Avoidance, Robotics & Automation Magazine, IEEE , vol.4, no.1, pp.23, 33, 1997

[44] D. Fox, W. Burgard, S. Thrun: The Dynamic Approach to Collision Avoidance Dept. Of Computer Science 3, University of Bonn, Germany, Carnegie Mellon University, Pittsburg.

[45] P. Ögren, N. E. Leonard: A Convergent Dynamic Window Approach to Obstacle Avoidance, IEEE Transaction on robotics, Vol. 21, No. 2, April 2005.

[46] Gu. Li, Y. Wu, W. Wei: Guided Dynamic Window Approach to Collision Avoidance in Troublesome Scenarios, College of Electric Engineering, Zhejiang University, Hangzhou, Proceedings of the 7th World Congress on Intelligent Control and Automation June 25–27, 2008, Chongqing, China.

[47] K. Rebai, O. Azouaoui, N. Ouadah: Bi-steerable Robot Navigation Using a Modified Dynamic Window Aproach, Centre de Développement des Technologies Avancées (CDTA), Cité 20 aout 1956, B.P.17, Baba Hassen, Alger, Algérie.

[48] D. Kiss, G. Tevesz: Advanced Dynamic Window based Navigation Approach using Model Predictive Control, Department of Automation and Applied Informatics, Budapest University of Technology and Economics, H-1117 Budapest, Hungary.

[49] B. Choi, B. Kim, E. Kim, K. W. Yang: A modified Dynamic Window Approach in crowded indoor environment for intelligent transport robot, Department of Electrical and Electronic Engineering, Yonsei University, Seoul, 120–749, Korea, Division for Applied Robot Technology, Korea Institute of Industrial Technology, Ansan, 426–791, Korea.

[50] E. Berger, T. Foote, B. Gerkey, K. Konolige: The Office Marathon: Robust Navigation in an Indoor Office Environment, Eitan Marder Eppstein, Willow Garage Inc., USA.

[51] S. Lee, J. H.Park: Dynamic Path-Following Using Temporaly Path Generator for Mobile Robots with Nonholonomic Constrains, Mechatornics La., Precision Mechanics Eng., Hanyang Univ., Seol, South Korea.

[52] F. Diaz del Rio, G. Jimenez, J.L. Sevillano, S. Vicente, A. Civit Balcells: A Generalization of Path Following for Mobile Robots, Facultad de Informatica, Univ. Sevilla, Spain.

[53] G. Indiveri, A. Nuchter, K. Lingermann: High Speed Differential Drive Mobile Robot Path Following Control With Bounded Wheel Speeds Commands, IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April, 2007.

[54] K.Komoriya, K.Tanie: Trajectory Design and Control of a Wheel-type Mobile Robot Using B-spline Curve, IEEE International Workshop on Intelligent Robots and Systems, Tsukiba, Japan, Spetember 4–6, 1989.

[55] P. Lepej, J. Maurer, S. Uran, G. Steinbauer: Dynamic arc fitting path follower for skid-steered mobile robots. International journal of advanced robotic systems, ISSN 1729-8814, 2015, vol. 12, str. 1–13, doi: 10.5772/61199. [COBISS.SI-ID 19010838]

[56] A. Jacoff, E. Messina, J. Evans: Experiences In Deploying Test Arenas For Autonomous Mobile Robots, Proceedings of the 2001 Performance Metrics for Intelligent Systems (PerMIS) Workshop, 2001

[57] RoboCup Wiki [Internet]. [updated on 28 March 2014]. Available from: en.wikipedia.org /wiki/RoboCup. Accessed 2014 April 7.

[58] Dynamic Arc Fitting path following algorritem [Internet]. http://wiki.ros.org/tedusar_daf_path_follower. Naloženo 18. 5. 2014.

[59] P. Lepej, A.Santamaria-Navarro, J. Solà: A flexible Hardware-in-the-loop architecture for UAVs, The 2017 International Conference on Unmanned Aircraft Systems, June 13–16, 2017, Miami Marriott Biscayne Bay, Miami, FL, USA.

[60] Prikaz delovanja navigacijskega sistema BLV [internet]: https://www.youtube.com/watch?v=zZEufTShyY8, dostop 24. 4. 2017.

[61] Hector BLV simulacija [internet]: http://wiki.ros.org/hector_quadrotor, dostop 24. 4. 2017.

[62] S. Kohlbrecher, J. Meyer, U. Klingauf,O. Stryk: A Flexible and Scalable SLAM System with Full 3D Motion Estimation, Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), pp. 155–160, November 1–5, 2011.

[63] OptiTrack sistem za lokalizacijo [internet]: http://optitrack.com/, dostop 24. 4. 2017.

[64] J. Q. Cui, S. Lai, X. Dong, P. Lui, Ben, M. C., Thong, H. L, B. M. Chen, T. H. Lee: Autonomous Navigation of UAV in Forest. In: International Conference on Unmanned Aircraft Systems (ICUAS), May 27–30, 2014. Orlando, FL, USA, 726–733.

[65] J. Cui, F. Wang, X. Dong, K. A. Zong Yao, B. M. Chen, T. H. Lee: Landmark Extraction and State Estimation for UAV Operation in Forest, In: Proceedings of the 32nd Chinese Control Conference, July 26–28, 2013, Xian, China, 5210–5215.

[66] C. Dornhege, A. Kleiner: A Frontier-Void-Based Approach for Autonomous Exploration in 3D, In: Proceedings of the 2011 IEEE International Symposium on Safety,Security and Rescue Robotics Kyoto,Japan,November 1–5, 2011, 351–356.

[67] K. Cesare, R. Skeele, Soo-H. Yoo, Y. Zhang, G. Hollinger: Multi-UAV Exploration with Limited Communication and Battery, In: 2015 IEEE International Conference on Robotics and Automation (ICRA) Washington State Convention Center Seattle, Washington, May 26–30, 2015, 2230–2235.

[68] C. Pravitra, G. Chowdhary, E. Johnson : A Compact Exploration Strategy for Indoor Flight Vehicles, In: 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC) Orlando, FL, USA, December 12–15, 2011, 3572–3577.

[69] J. P. Laumond, P. E. Jacobs, M. Taix, R. M. Murray: A Motion Planner for Nonholonomic Mobile Robots, IEEE Transaction on Robotics and Automation, Vol. 10, NO. 5., 1994

[70] A. A. Awwalur Rizqi, A. I. Cahyadi, T. B. Adji: Path Planning and Formation Control via Potential Function for UAV Quadrotor. In: 2014 International Conference on Advanced Robotics and Intelligent Systems (ARIS 2014), June 6–8, 2014, Nat'l Taiwan Univ. of Sci. and Tech., Taipei, Taiwan, R.O.C., 165–170.

[71] D. M. Rivera, F. A. Prieto, R. Ramirez: Trajectory Planning for UAVs in 3D Environments Using a Moving Band in Potential Sigmoid Fields. In: Brazilian Robotics Symposium and Latin American Robotics Symposium, 2012, 115–119.

[72] G. Mester, A. Rodic.: Navigation of an Autonomous Outdoor Quadrotor Helicopter. In: University of Szeged, Faculty of Engineering, Robotics Laboratory, Szeged, Hungary, University of Belgrade, Institute Mihajlo Pupin, Robotics Laboratory, Belgrade, Serbia.

[73] A. Akhtar, S. L. Waslander , C. Nielsen: Path following for a quadrotor using dynamic extension and transverse feedback linearization. In: 51st IEEE Conference on Decision and Control December 10–13, 2012. Maui, Hawaii, USA, 3551–3556.

[74] Podjetje ASCTEC Pelican: http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-pelican/, dostop 24. 4. 2017

[75] Youtube posnetek letečega robota v simulaciji: https://www.youtube.com/watch?v=tsTfZm3CcNM, dostop 25. 4. 2017.

[76] J. Bongard: Probabilistic Robotics. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. (2005, MIT Press.) 647 pages, in Artificial Life, vol. 14, no. 2, pp. 227-229, April 2008. doi: 10.1162/artl.2008.14.2.227

[77] A. Martinez, E. Fernndez: Learning ROS for Robotics Programming, Packt Publishing ©2013

[78] B. Gerkey, W. Smart, M. Quigley: Programming Robots with ROS, A Practical Introduction to the Robot Operating System, O'Reilly Media, December 2015

[79] A. Zelinsky: Learning OpenCV---Computer Vision with the OpenCV Library (Bradski, G.R. et al.; 2008)[On the Shelf], IEEE Robotics and Automation Magazine, September 2009

[80] P. Corke: Robotics, Vision and Control, Fundamental Algorithms in MATLAB, Robotics and Automation, Springer-Verlag Berlin Heidelberg, 2011

[81] G. Cook: Mobile Robots: Navigation, Control and Remote Sensing, Wiley Online Library, 24 OCT 2011

[82] A. Kelly: Mobile Robotics, Mathematics, Models, and Methods, Carnegie Mellon University, Pennsylvania, June 2014

[83] J. Y. Wong: Theory of Ground Vehicles, 4th Edition, Wiley, August 2008

[84] K. Iagnemma, S. Dubowsky: Mobile Robots in Rough Terrain, Estimation, Motion Planning, and Control with Application to Planetary Rovers, Springer, 2004

[85] P. Lamon: 3D-Position Tracking and Control for All-Terrain Robots, Springer, 2008

[86] R. Isermann, Fault-Diagnosis Applications, Model-Based Condition Monitoring: Actuators, Drives, Machinery, Plants, Sensors, and Fault-tolerant Systems, Springer, 2011

[87] Opisi RoboCup Rescue Ekip: http://www.robocup2016.org/en/symposium/team-description-papers/rescue-robot/, dostop 25. 11. 2017

8 ŽIVLJENJEPIS

Peter Lepej je leta 2009 diplomiral iz smeri Mehatronika na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru pod mentorstvom red. prof. dr Rika Šafariča. Pod mentorstvom dr. Jurija Rakuna se je seznanil z avtonomnim poljskim robotom in vrsto let sodeloval kot član ekipe na mednarodnih tekmovanjih FieldRobot Event, kjer je dosegal tudi vidne rezultate. Po končanem študiju se je zaposlil v podjetju AHA EMMI, d. o. o., kjer je tudi opravil obvezno prakso. V podjetju je deloval kot razvojni inženir in vodja projektov ter opravljal zahtevnejše vzdrževanje. Po slabih dveh letih mu je dr. Suzana Uran ponudila sodelovanje v triletnem projektu SI-AT TEDUSAR, pričel je z doktorskim študijem in se zaposlil na Fakulteti za elektrotehniko, računalništvo in informatiko. Pri projektu TEDUSAR se je ukvarjal z reševalno mobilno robotiko oziroma avtonomnim premikanjem mobilnih robotov v neurejenih okoljih. Po zaključenem triletnem projektu na fakulteti je kandidat pridobil državno štipendijo Javnega sklada Republike Slovenije za razvoj kadrov in štipendij za delo na Institutu za industrijsko robotiko v Barceloni, v Španiji. Tam je nadaljeval z delom na navigacijskih sistemih za avtonomne mobilne robote. Po končanem delu v tujini se je zaposlil v lastnem podjetju VISTION, d. o. o., katerega glavna dejavnost je izdelava aplikacij strojnega vida v industriji.

Ime in priimek: Peter Lepej

Rojen: 18. 4. 1985

Oče: Silvo Lepej

Mati: Rozika Lepej

Šolanje:	1992–2000	Osnovna šola Črešnjevec, Slovenska Bistrica
	2000–2004	Srednja elektro-računalniška šola Maribor, tehnična gimnazija, Maribor
	2004–2009	Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Mehatronika, Maribor
	2012–2017	Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, doktorski študijski program Elektrotehnika, Maribor

Zaposlitev:	2010–2011	AHA EMMI, Slovenska Bistrica
	2011–2014	Univerza v Mariboru, Fakulteta za elektroteniko, računalništvo in informatiko, Maribor
	april 2015–julij 2015	Institut za industrijsko robotiko, Barcelona, Španija
	avgust 2015	Univerza v Mariboru, Fakulteta za kmetijstvo in biosistemske vede, Hoče, Maribor
	september 2015–	VISTION d.o.o., Ljubljana

9 BIBLIOGRAFIJA

9.1 IZVIRNI ZNANSTVENI ČLANEK

1. LEPEJ, Peter, RAKUN, Jurij. Simultaneous localisation and mapping in a complex field environment. *Biosystems engineering*, ISSN 1537-5110. [Print ed.], October 2016, vol. 150, str. 160-169, ilustr. [COBISS.SI-ID 4202540], [JCR, SNIP, WoS do 26. 3. 2017: št. citatov (TC): 1, čistih citatov (CI): 1, Scopus do 21. 2. 2017: št. citatov (TC): 1, čistih citatov (CI): 1]

2. LEPEJ, Peter, MAURER, Johannes, URAN, Suzana, STEINBAUER, Gerald. Dynamic arc fitting path follower for skid-steered mobile robots. *International journal of advanced robotic systems*, ISSN 1729-8814, 2015, vol. 12, str. 1-13. https://dk.um.si/IzpisGradiva.php?id=66840, doi: 10.5772/61199. [COBISS.SI-ID 19010838], [JCR, SNIP, WoS do 24. 10. 2015: št. citatov (TC): 0, čistih citatov (CI): 0, Scopus do 28. 4. 2017: št. citatov (TC): 1, čistih citatov (CI): 1]

3. KLANČNIK, Simon, MEDVEŠEK, Aleš, LEPEJ, Peter, BENKIČ, Karl. Algoritem za izločanje laserske črte pri zaznavanju ovir s strukturirano svetlobo = Laser line elimination algorithm for obstacle detection using structures light. *Ventil : revija za fluidno tehniko in avtomatizacijo*, ISSN 1318-7279, avg. 2007, letn. 13, 4, str. 238-245. [COBISS.SI-ID 11568918]

9.2 OBJAVLJENI ZNANSTVENI PRISPEVEK NA KONFERENCI

4. LEPEJ, Peter, SANTAMARIA-NAVARRO, Angel, SOLÀ, Joan. A flexible hardware-in-the-loop architecture for UAVs. V: *2017 International Conference on Unmanned Aircraft Systems (ICUAS), June 13-16, 2017, Miami, Florida, USA*. [S. l.]: IEEE. 2017, str. 1751-1756.http://ieeexplore.ieee.org.ezproxy.lib.ukm.si/stamp/stamp.jsp?tp=&arnumber=7991330. [COBISS.SI-ID 20719126]

5. LEPEJ, Peter, LAKOTA, Miran, RAKUN, Jurij. Robotic real-time 3D object reconstruction using multiple laser range finders. V: TAYLOR, James (ur.), et al. *Precision Agriculture '17 : papers presented at the 11th European Conference on Precision Agriculture (ECPA 2017), John McIntyre Centre, Edinburgh, UK, [July 16-20 2017]*, (Advances in Animal Biosciences, ISSN 2040-4700, Vol. 8, Iss. 2). Cambridge: Cambridge University Press. 2017, str. 183-188, ilustr. [COBISS.SI-ID 4340268]

6. LAKOTA, Miran, BERK, Peter, RAKUN, Jurij, LEPEJ, Peter, KRANER, Jože, PURGAJ, Miran, ŠLAMBERGER, Zvonko, JAGODIČ, Uroš. Cornstar robot. V: VÁZQUEZ ARELLANO, Manuel (ur.), GRIEPENTROG, Hans W. (ur.). *Proceedings of the 12th Field Robot Event 2014, Bernburg-Strenzfeld/Germany, June 17th-19th, 2014 : Conducted in conjunction with the DLG-Feldtage/DLG Field Days*. Stuttgart: University of Hohenheim, Instrumentation & Test Engineering. 2015, str. 66-70, ilustr. [COBISS.SI-ID 3938860]

7. ZAMAN, Safdar, LEPEJ, Peter. ROS-based diagnostic board for detecting and repairing hardware faults in autonomous mobile robots. V: *2014 IEEE International Conference on robotics and emerging allied technologies in engineering (iCREATE) : proceedings*. [S. l.]: IEEE. cop. 2014, str. 95-102, doi: 10.1109/iCREATE.2014.6828347. [COBISS.SI-ID 17978646], [WoS do 9. 4. 2017: št. citatov (TC): 0, čistih citatov (CI): 0, Scopus do 27. 11. 2014: št. citatov (TC): 0, čistih citatov (CI): 0]

8. LEPEJ, Peter, LAKOTA, Miran, RAKUN, Jurij. Crop line based field robot navigation. V: GONZALES-DE-SANTOS, Pablo (ur.), RIBEIRO, Angela (ur.). *RHEA-2014 : new trends in mobile robotics, perception and actuation for agriculture and forestry*. [Madrid: RHEA project. 2014], str. 377-385, ilustr. [COBISS.SI-ID 3720748]

9. ZAMAN, Safdar, STEINBAUER, Gerald, MAURER, Johannes, LEPEJ, Peter, URAN, Suzana. An integrated model-based diagnosis and repair architecture for ROS-based robot systems. V: *ICRA 2013*, 2013 IEEE International Conference on Robotics and Automation, May 6-10, 2013 Karslruhe, Germany. [S. l.]: IEEE. cop. 2013, str. 482-489, doi: 10.1109/ICRA.2013.6630618. [COBISS.SI-ID 17645334], [Scopus do 26. 6. 2017: št. citatov (TC): 13, čistih citatov (CI): 12]

10. ZAMAN, Safdar, LEPEJ, Peter. MBD counterpart diagnostic board : its hardware, protocol, software, simulator and application. V: NEMEC, Bojan (ur.), ŽLAJPAH, Leon (ur.). *Proceedings*, 22nd International Workshop on Robotics in Alpe-Adria-Danube Region [also] RAAD 2013, September 11-13, Portorož, Slovenia. 1st ed. Ljubljana: Jožef Stefan Institute. 2013, str. 86-93. [COBISS.SI-ID 17242390]

11. LEPEJ, Peter, MAURER, Johannes, STEINBAUER, Gerald, URAN, Suzana. Vožnja mobilnega robota z gosenicami po razgibani podlagi in stopnicah = Driving tracked mobile robot over complex enviromet and stairs. V: ZAJC, Baldomir (ur.), TROST, Andrej (ur.). *Zbornik dvaindvajsete mednarodne Elektrotehniške in računalniške konference ERK 2013, 16.-18. september 2013, Portorož, Slovenija*, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., ISSN 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE. 2013, zv. B, str. 99-102. [COBISS.SI-ID 17175830]

12. LEPEJ, Peter, MAURER, Johannes, STEINBAUER, Gerald, URAN, Suzana, ZAMAN, Safdar. An integrated diagnosis and repair architecture for ROS-based robot systems. V: DEARDEN, Richard (ur.), SNOOKE, Neal (ur.). *Proceedings of the 23th International Workshop on Principles of Diagnosis (DX-2012) : Malvern, UK, July 31 to August 3, 2012*. [S. l.: s. n. 2012], str. 35-42. [COBISS.SI-ID 16256278]

13. URAN, Suzana, STEINBAUER, Gerald, LEPEJ, Peter, MAURER, Johannes. Roboti za reševanje po potresu. V: BOHANEC, Marko (ur.), et al. *Zbornik 15. mednarodne multikonference Informacijska družba - IS 2012, 8.-12. oktober 2012, Ljubljana, Slovenia : zvezek A = Proceedings of the 15th International Multiconference Information Society - IS 2012, October 8th-12th, 2012, [Ljubljana, Slovenia] : volume A*, (Informacijska družba, ISSN 1581-9973). Ljubljana: Institut Jožef Stefan. 2012, str. 333-336. [COBISS.SI-ID 16349974]

14. LEPEJ, Peter, MAURER, Johannes, STEINBAUER, Gerald, URAN, Suzana. Analysis of laser sensor system for a rescue robot. V: ZAJC, Baldomir (ur.), TROST, Andrej (ur.). *Zbornik enaindvajsete mednarodne Elektrotehniške in računalniške konference ERK 2012, 17.-19. september 2012, Portorož, Slovenija*, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., ISSN 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE. 2012, zv. B, str. 127-130, ilustr. [COBISS.SI-ID 16306966]

15. RAKUN, Jurij, BERK, Peter, PARIPOVIČ, Tomaž, LEPEJ, Peter, LAKOTA, Miran, STAJNKO, Denis. Selektivni nanos fitofarmacevtskih sredstev z uporabo pametnega robota. V: KOŠUTIĆ, Silvio (ur.). *Aktualni zadaci mehanizacije poljoprivrede : zbornik radova 38. Međunarodnog simpozija iz područja mehanizacije poljoprivrede, Opatija, 22. - 26. veljače 2010 = [Actual tasks on agricultural engineering : proceedings of the 38. International Symposium on Agricultural Engineering, Opatija, Croatia, 22.-26. February 2010*. Zagreb: Sveučilište u Zagrebu, Agronomski fakultet, Zavod za mehanizaciju poljoprivrede. 2010, str. 191-199. [COBISS.SI-ID 2909740]

16. BERK, Peter, LEPEJ, Peter, PARIPOVIĆ, Tomaž, RAKUN, Jurij, LAKOTA, Miran. CornStar. V: HARMS, Hans H., LANG, Thorsten, SCHATTENBERG, Jan. *Proceedings of the 8th Field Robot Event 2010, Braunschweig/Germany, June 11-13, 2010 : Prototype contest*. Braunschweig, Germany: Institute for Agricultural Machinery and Fluid Power. 2010, str. 42-55. [COBISS.SI-ID 3213100]

17. LAKOTA, Miran, BERK, Peter, RAKUN, Jurij, LEPEJ, Peter, PARIPOVIČ, Tomaž. The Autonomous Robot - Corn Star project. V: *Proceedings of the 7th Field Robot Event 2009, Wageningen, July 6 and 7, 2009*. Wageningen: Wageningen University Farm Technology Group. 2009, str. 25-32. [COBISS.SI-ID 2947372]

18. MEDVEŠEK, Aleš, LEPEJ, Peter. Zaznavanje ovir s strukturirano svetlobo - lasersko črto = Obstacle detection with structured light - laser line. V: TOVORNIK, Boris (ur.), MUŠKINJA, Nenad (ur.). *Zbornik pete konference Avtomatizacija v industriji in gospodarstvu, 11. in 12. april 2007, Maribor, Slovenija*. Maribor: Društvo avtomatikov Slovenije. 2007, str. 305-310. [COBISS.SI-ID 11228950]

9.3 OBJAVLJENI STROKOVNI PRISPEVEK NA KONFERENCI

19. LEPEJ, Peter, MIHELJ, Rok. Preverjanje ram pokrovčkov na brizgalnem stroju s strojnim vidom. V: ŠVETAK, Darko (ur.). *Vir znanja in izkušenj za stroko : zbornik foruma*, [8.] industrijski forum IRT, Portorož, 6. in 7. junij 2016. Škofljica: Profidtp. 2016, str. 201-204. [COBISS.SI-ID 19650326]

20. LEPEJ, Peter, BERK, Peter, RAKUN, Jurij, LAKOTA, Miran. Prednosti in slabosti digitalne obdelave slik na poljskem robotu. V: POTOČNIK, Božidar (ur.). *ROSUS 2013 : računalniška obdelava slik in njena uporaba v Sloveniji 2013 : zbornik 8. strokovne konference, Maribor, 21. marec 2013*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, Inštitut za računalništvo. 2013, str. 39-47. [COBISS.SI-ID 3492140]

21. LEPEJ, Peter, PARIPOVIČ, Tomaž, RAKUN, Jurij. Avtomatiziran poljedelski stroj. V: ZAJC, Baldomir (ur.), TROST, Andrej (ur.). *Zbornik sedemnajste mednarodne Elektrotehniške in računalniške konference ERK 2008, 29. september - 1. oktober 2008, Portorož, Slovenija*, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., ISSN 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE. 2008, str. 369-370. [COBISS.SI-ID 2701868]

9.4 OBJAVLJEN POVZETEK ZNANSTVENEGA PRISPEVKA NA KONFERENCI

22. KOHLBRECHER, Stefan, PETERSEN, Karen, STEINBAUER, Gerald, MAURER, Johannes, LEPEJ, Peter, URAN, Suzana, VENTURA, Rodrigo, DORNHEGE, Christian, HERTLE, Andreas, SHEH, Raymond, PELLENZ, Johannes. Community-driven development of standard software modules for search and rescue robots. V: *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR): College Station, TX, USA, November 5-8, 2012*. [Piscataway, N.J.]: IEEE. cop. 2012.,, str. 1-2, doi: 10.1109/SSRR.2012.6523917. [COBISS.SI-ID 17652246], [Scopus do 25. 4. 2017: št. citatov (TC): 5, čistih citatov (CI): 5]

9.5 SAMOSTOJNI ZNANSTVENI SESTAVEK ALI POGLAVJE V MONOGRAFSKI PUBLIKACIJI

23. LEPEJ, Peter, RAKUN, Jurij. Uporaba avtonomnih mobilnih robotov v kmetijstvu. V: MUNIH, Marko (ur.). *SI robotika*. Ljubljana: Slovenska matica. 2017, str. 77-88, ilustr. [COBISS.SI-ID 11702868]

24. LEPEJ, Peter, LAKOTA, Miran, RAKUN, Jurij. Simultanious localization, maping and scene reconstruction. V: KATALINIĆ, Branko (ur.). *DAAAM International scientific book 2016*, (DAAAM International scientific book, ISSN 1726-9687). Vienna: DAAAM International Vienna. 2016, str. 69-76, ilustr. [COBISS.SI-ID 4260652]

9.6 DIPLOMSKO DELO

25. LEPEJ, Peter. *Uporaba računalniškega vida za nadzor in upravljanje mobilnih platform : diplomsko delo univerzitetnega študijskega programa*. Maribor: [P. Lepej], 2009. VIII, 78 f., ilustr. http://dkum.uni-mb.si/Dokument.php?id=10474. [COBISS.SI-ID 13681686]

9.7 KONČNO POROČILO O REZULTATIH RAZISKAV

26. MAURER, Johannes, STEINBAUER, Gerald, LEPEJ, Peter, URAN, Suzana. *TeDUSAR white book - state of the art in search and rescue robots*. [S. l.: s. n., 2014]. http://www.tedusar.eu/files/tedusar_white_book_v_1_0.pdf, http://www.tedusar.eu/cms/sl/first_responder/tedusar-whitebook. [COBISS.SI-ID 20609558]

9.8 ELABORAT, PREDŠTUDIJA, ŠTUDIJA

27. LEPEJ, Peter, URAN, Suzana. *Path planning in 3D environment : project TEDUSAR*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2014. 28 f., ilustr. [COBISS.SI-ID 20608278]

28. LEPEJ, Peter, URAN, Suzana. *Analiza pristopa dinamičnega okna za načrtovanje poti*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2013. 55 f., ilustr. [COBISS.SI-ID 20607766]

29. LEPEJ, Peter, URAN, Suzana. *Prepoznavanje tal pred robotom z nevronsko mrežo*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2013. 39 f., ilustr. [COBISS.SI-ID 20608022]

30. LEPEJ, Peter, URAN, Suzana. *Modeli mobilnega robota*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2012. 33 f., ilustr. [COBISS.SI-ID 20606998]

31. LEPEJ, Peter, URAN, Suzana. *Primerjava postopkov za kartiranje z mobilnim robotom*. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2012. 27 f., ilustr. [COBISS.SI-ID 20607254]

9.9 PRISPEVEK NA KONFERENCI BREZ NATISA

32. URAN, Suzana, LEPEJ, Peter. *Avtonomni reševalni robot = Autonomous rescue robot : predavanje na Mednarodnem sejmu za avtomatizacijo, robotiko in mehatroniko, IFAM - Business forum, 31. 01. 2014, Celje*. 2014. [COBISS.SI-ID 17622550]

9.10 NERAZPOREJENO

33. VEHOVAR, Sebastijan, LEPEJ, Peter. *Večfunkcijska digitalna ura budilka : elektronika : raziskovalna naloga*. Maribor: Srednja elektro-računalniška šola, 2003. 43 f., graf. prikazi. [COBISS.SI-ID 12870408]

34. MARČIČ, Aljaž, LEPEJ, Peter, MOČNIK, Dejan. *Stari grad : raziskovalno področje zgodovina : raziskovalna naloga*. Maribor: SERŠ, 2002. 30 f., ilustr. [COBISS.SI-ID 11653128]

35. VEHOVAR, Sebastijan, LEPEJ, Peter. *Svetlobni efekti*. Maribor: SERŠ, 2001. 36 f., graf. prikazi. [COBISS.SI-ID 10757128]

Izpis bibliografskih enot:	vse bibliografske enote
Izbrani format bibliografske enote:	ISO 690
Razvrščanje bibliografskih enot:	tipologija, leto - padajoče, naslov
Vir bibliografskih zapisov:	Vzajemna baza podatkov COBISS.SI/COBIB.SI, 22. 8. 2017

Datum ažuriranja baze JCR (letno): 10. 10. 2016

Datum ažuriranja baze SNIP (letno): 30. 8. 2016

Datum ažuriranja povezav med zapisi v COBIB.SI in WoS ter števila citatov (dnevno): 22. 8. 2017

Datum ažuriranja povezav med zapisi v COBIB.SI in Scopus ter števila citatov (dnevno): 22. 8. 2017

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

IZJAVA DOKTORSKEGA KANDIDATA

Podpisani-a PETER LEPEJ,

vpisna številka E9504697.

izjavljam,

da je doktorska disertacija z naslovom
SISTEMI IN KRMILJENJE AVTONOMNIH MOBILNIH ROBOTOV V NEUREJENIH PROSTORIH

- rezultat lastnega raziskovalnega dela,
- da predložena disertacija v celoti ali v delih ni bila predložena za pridobitev kakršnekoli izobrazbe po študijskem programu druge fakultete ali univerze,
- da so rezultati korektno navedeni in
- da nisem kršil-a avtorskih pravic in intelektualne lastnine drugih.

Podpis doktorskega-e kandidata-ke:

Peter Lepej

Obrazec RŠZ

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

IZJAVA O OBJAVI ELEKTRONSKE VERZIJE DOKTORSKE DISERTACIJE IN OSEBNIH PODATKOV, VEZANIH NA ZAKLJUČEK ŠTUDIJA

Ime in priimek doktoranda-ke: PETER LEPEJ
Vpisna številka: E9504697
Študijski program: ELEKTROTEHNIKA
Naslov doktorskega dela:

SISTEMI IN KRMILJENJE AVTONOMNIH MOBILNIH ROBOTOV V NEUREJENIH PROSTORIH

Mentor-ica: DOC. DR. SUZANA URAN
Somentor-ica:

Podpisani soglašam z objavo doktorske disertacije v Digitalni knjižnici Univerze v Mariboru.

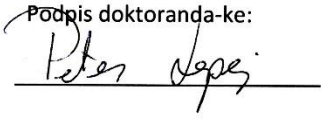
Tiskana verzija doktorske disertacije je istovetna elektronski verziji, ki sem jo oddal-a v Digitalno knjižnico Univerze v Mariboru.

Podpisani-a hkrati izjavljam, da dovoljujem objavo osebnih podatkov, vezanih na zaključek študija (ime, priimek, leto in kraj rojstva, datum diplomiranja, naslov diplomskega dela) na spletnih straneh in v publikacijah Univerze v Mariboru.

Datum in kraj:

15.5.2017, PRETREŽ

Podpis doktoranda-ke:



Obrazec RŠZ

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

IZJAVA KANDIDATOVEGA MENTORJA O USTREZNOSTI DOKTORSKE DISERTACIJE

Podpisani-a DOC. DR. SUZANA URAN, mentor-ica doktorskemu-i kandidatu-ki,
izjavljam, da je doktorska disertacija z naslovom
SISTEMI IN KRMILJENJE AVTONOMNIH MOBILNIH ROBOTOV V NEUREJENIH PROSTORIH,
ki jo je izdelal-a doktorski-a kandidat-ka PETER LEPEJ,
v skladu z odobreno temo, Pravilnikom o pripravi in zagovoru doktorske disertacije ter mojimi navodili in predstavlja izviren prispevek k razvoju znanstvene discipline.

Datum in kraj:
Maribor, 25.4.2017

Podpis mentorja-ice:
Uran Suzana

Obrazec RŠZ