Realizacija SLAM sustava uz algoritam otvorenog koda i LiDAR čvrstog stanja

Franković, Nino

SVEUČILIŠTE U ZAGREBU
GEODETSKI FAKULTET

Nino Franković

REALIZACIJA SLAM SUSTAVA UZ ALGORITAM OTVORENOG KODA I LiDAR ČVRSTOG STANJA

Diplomski rad

Zagreb, 2022.

Nino Franković ◆ DIPLOMSKI RAD ◆ 2022.

SVEUČILIŠTE U ZAGREBU

GEODETSKI FAKULTET

Nino Franković

REALIZACIJA SLAM SUSTAVA UZ ALGORITAM OTVORENOG KODA I LiDAR ČVRSTOG STANJA

Diplomski rad

Zagreb, 2022.

SVEUČILIŠTE U ZAGREBU

GEODETSKI FAKULTET

Na temelju članka 19. Etičkog kodeksa Sveučilišta u Zagrebu i Odluke br. 1_349_11 Fakultetskog vijeća Geodetskog fakulteta Sveučilišta u Zagrebu, od 26. 10. 2017. godine (klasa: 643-03/16-07/03), uređena je obaveza davanja „Izjave o izvornosti“ diplomskog rada koji se vrednuju na diplomskom studiju geodezije i geoinformatike, a u svrhu potvrđivanja da je rad izvorni rezultat rada studenata te da taj rad ne sadržava druge izvore osim onih koji su u njima navedeni.

IZJAVLJUJEM

Ja, **Nino Franković** (JMBAG: 0007178248), rođen dana 15. 06. 1995. u Rijeci, izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi tog rada nisam koristio drugim izvorima osim onih koji su u njemu navedeni.

U Zagrebu, dana _____ _____

Potpis studenta

I. AUTOR	
Ime i prezime:	Nino Franković
Datum i mjesto rođenja:	15. lipnja 1995., Rijeka, Republika Hrvatska
II. DIPLOMSKI RAD	
Naslov:	Realizacija SLAM sustava uz algoritam otvorenog koda i LiDAR čvrstog stanja
Broj stranica:	63
Broj tablica:	5
Broj slika:	38
Broj bibliografskih podataka:	22 + 40 URL-a
Ustanova i mjesto gdje je rad izrađen:	Geodetski fakultet Sveučilišta u Zagrebu
Mentor:	doc. dr. sc. Loris Redovniković
III. OCJENA I OBRANA	
Datum zadavanja teme:	14. 01. 2020.
Datum obrane rada:	11.02.2022.
Sastav povjerenstva pred kojim je branjen diplomski rad:	doc. dr. sc. Loris Redovniković
	prof. dr. sc. Đuro Barković
	prof. dr. sc. Mladen Zrinjski

Popis i objašnjenje kratica korištenih u radu

3D	Trodimenzionalno
API	Aplikacijsko programsko sučelje (engl. *Application Programming Interface*)
FOV	Vidno polje (engl. *Field Of View*)
GNU	Gnu nije Unix (engl. *GNU's Not Unix*)
GUI	Grafičko korisničko sučelje (engl. *Graphical User Interface*)
GPS	Globalni pozicijski sustav (engl. *Global Positioning System*)
IMU	Inercijalna mjerna jedinica (engl. *Inertial Measurment Unit*)
INS	Inercijalni navigacijski sustav (engl. *Inertial Navigation System*)
LiDAR	Svijetlosno zamjećivanje i klasifikacija (engl. *Light Detection and Ranging*)
LOAM	Livox odometrija i kartiranje (engl. *Livox Odometry And Mapping*)
MLS	Mobilno lasersko skeniranje/skener *(engl. Mobile Laser Scanning/Scanner*)
PCD	Oblak točaka - format (engl. *PointCloud*)
PCL	Biblioteka oblaka točaka (engl. *Point Cloud Library*)
PLY	Format podataka poligona (engl. *Polygon File Format*)
ROS	Robotski operacijski sustav (engl. *Robot Operating System*)
SDK	Paket za razvoj softvera (engl. *Software Development Kit*)
SLAM	Simultana lokalizacija i kartiranje (engl. *Simultaneous Localization And Mapping*)
TLS	Terestričko lasersko skeniranje/skener (engl. *Tereestrial Laser Scanning/Scanner*)

Zahvala

Zahvaljujem mentoru doc. dr. sc. Lorisu Redovnikoviću koji me uveo i zainteresirao za područje primjene laserskih uređaja, te me strpljivo i marljivo svojim velikim znanjem vodio kroz izradu ovog rada.

Zahvaljujem svim svojim prijateljima i kolegama koji su uvijek bili uz mene, a posebno zahvaljujem dragom prijatelju i voditelju Antunu Jakopecu čija pomoć mi je uvijek bila na raspolaganju na fakultetu, a društvo van fakulteta.

Na kraju, veliko hvala cijeloj mojoj obitelji, rodbini i najviše djevojci na razumijevanju, ljubavi i podršci tijekom svih ovih godina studiranja.

Realizacija SLAM sustava uz algoritam otvorenog koda i LiDAR čvrstog stanja

Sažetak**:** *LiDAR je mjerni uređaj koji pomoću laserskih zraka opaža objekte i pojave u svojoj okolini. Na ovaj način dobiva se veliki obujam informacija o prostoru u znatno kraće vrijeme no što je to bilo potrebno kod prethodnih metoda izmjere. Međutim, kako bi podaci dobiveni laserskim skenerima bili korisni, potrebno je dobro poznavati problematiku samog načina nastanka, obrade i pohrane trodimenzionalnog oblaka točaka. Ako se tome doda i problem kretanja, te potreba za praćenjem lokacije uređaja za svo vrijeme snimanja, dolazi se do kompleksnog sustava koji uključuje razne mjerne sustave i softverska rješenja. Kartiranje okoliša pri kojem se svo vrijeme vrši i lokalizacija naziva se SLAM i zbog svoje kompleksnosti cijena gotovih komercijalnih SLAM sustava je vrlo visoka. Cilj ovog rada je upogoniti SLAM sustav na način da se koristi niskobudžetni LiDAR čvrstog stanja, a kao softverska podrška, otvoreni algoritam koji je svima dostupan na GitHub platformi. Korišteni LiDAR uređaj je Livox Mid-100, a korišteni algoritam kao softverska podrška je razvijen na sveučilištu u Hong Kongu pod nazivom LOAM Livox. Eventualna mogućnost budućeg korištenja ovakvih sustava u geodetske ili slične svrhe snimanja okoliša zasigurno bi približila lasersku tehnologiju širem broju korisnika. Korištenje cjenovno prihvatljivijih novih tehnologija također bi pozitivno utjecalo na modernizaciju struke općenito.*

Ključne riječi: *LiDAR čvrstog stanja, SLAM, ROS, LOAM*

Realization of SLAM system with open-source algorithm and solid-state LiDAR

Abstract: *LiDAR is a measuring device that uses laser beams to record objects and phenomena in its environment. In this way, a large amount of spatial information is obtained in a considerably shorter time than was required with previous measurement methods. However, in order for the data obtained by laser scanners to be useful, it is necessary to have an understanding of the issues of the very way of creating, storing and processing the so-called 3D point clouds. Adding the problem of movement, as well as the need to monitor the location of the device for the entire recording time, results is a complex system that includes various measuring devices and software solutions. Mapping the environment while simultaneously preforming localizations is named SLAM, and due to its complexity the price of ready-made commercial SLAM systems is very high. The aim of this study is to power up the SLAM system in a way that uses a low-budget solid-state LiDAR, and as software support, an open type algorithm that is publicly available via the GitHub platform. The LiDAR device used is the Livox Mid-100, while the algorithm used is LOAM Livox, developed at the University of Hong Kong. The possible future use of such systems for geodetic or similar purposes of environmental mapping would certainly bring laser technology closer to a wider number of users. The use of more affordable new technologies would also have a positive impact on the modernization of the profession in general.*

Keywords: *Solid-state LiDAR, SLAM, ROS, LOAM*

SADRŽAJ

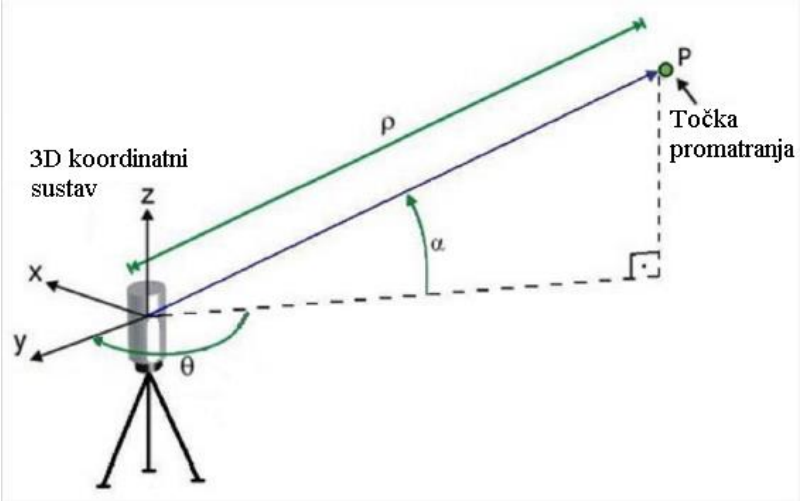
1. UVOD

U sve užurbanijem svijetu najveća prednost i vrlina postaje brzina, a vrijeme najveće bogatstvo. Kako bi se mogao pratiti takav užurbani život, što privatno, što poslovno, potrebno se prilagoditi. Nove tehnologije pojavljuju se u sve kraćim vremenskim intervalima, a konstantno učenje, unaprjeđivanje svojih znanja i implementacija istih u nove projekte i ideje neminovno je kako bi si pojednostavili i poboljšali življenje. Jedan od načina kojim se može u kratkom roku izmjeriti pojedini objekt ili područje i dobiti vrlo vjeran trodimenzionalni prikaz izmjerenog je lasersko skeniranje. Međutim, laserski skeneri su prilično skupi uređaji i njihov daljinski doseg mjerenja je relativno kratak. Drugi način za dobivanje, također vrlo preglednih i preciznih, trodimenzionalnih modela otvorenog ili zatvorenog prostora je fotogrametrija. Najveća mana kod ove metode je činjenica da je za dobre fotogrametrijske modele potrebno jako dobro osvjetljenje svih snimanih dijelova. Zbog toga fotogrametrijsko snimanje nije posve neovisno te već pokoja sjena kvari rezultate i prouzrokuje šum, dok je izmjera noću praktički nemoguća. Prateći rastući razvoj tržišta LiDAR-a uočljiva je sve veća prisutnost LiDAR-a čvrstog stanja (engl. *Solid State LiDAR*) koji svojim osobinama zadovoljavaju upravo ono što današnjica traži, a to je brzina prikupljanja podataka, iskoristivost krajnjeg proizvoda i rezultata i cijena samog uređaja i kompletnog sustava. Ova vrsta LiDAR-a prvenstveno se razvija za implementaciju u robote i autonomna vozila kako bi se oni mogli orijentirati u prostoru i u realnom vremenu pratiti događaje u svojoj neposrednoj okolini, te na njih reagirati. Međutim, svojim karakteristikama obećavaju mnogo i kao instrumenti za geodetske izmjere, kartiranja i izradu raznih 3D modela. Uz to, ideja ovog rada je uspostaviti SLAM sustav. To je sustav koji pomoću LiDAR-a prikuplja podatke i istovremeno se orijentira u prostoru i kartira okolinu. Mogućnost laserskog snimanja u pokretu otvara mogućnost snimanja većih područja, snimanje nekog objekta sa svih strana (nema potrebe za više stajališta i naknadnim spajanje podataka), te bitno skraćuje vrijeme potrebno za izvršenje zadatka. Domet snimanja od 260 metara, frekvencija od 300 000 točaka u sekundi i cijena preko 10 puta niža od vrijednosti sličnih komercijalnih sustava dovoljan su razlog za krenuti u dublje proučavanje ovog sustava u svrhu procijene njegovog potencijala i iskoristivosti u geodetskim mjerenjima.

2. VRSTE TERESTRIČKIH LASERSKIH SKENERA

LiDAR (engl. *Light Detection and Ranging*): svjetlosno zamjećivanje i klasifikacija je optički mjerni instrument koji odašilje laserske zrake koje se odbijaju od vrlo sitnih čestica raspršenih u Zemljinoj atmosferi (aerosola, oblačnih kapljica i drugo) i potom registriraju u optičkom prijamniku (obično teleskopu). Većina ove tehnologije radi na način da odbija pulsirajući laser od rotirajućeg zrcala na udaljene objekte, a potom bilježi vrijeme povratka reflektiranog impulsa. Drugi naziv za LiDAR je optički radar (engl. *Light radar*) i laserski radar (URL 1).

Terestrički laserski skener radi na principu sličnom kao mjerne stanice odnosno na principu mjerenja vertikalnog i horizontalnog kuta te duljine (Slika 2.1), ali razlika je u brzini izvođenja mjerenja, količini podataka i samim aplikacijama. Laserski skener izvodi operacije mjerenja samostalno i neselektivno (Staiger, 2009). Kod mjerenja s laserskim skenerom se ne vizira točka od interesa koja je reprezentativna za traženi objekt, već se snimaju sve dostupne točke objekta koje sačinjavaju trodimenzionalni oblak točaka (engl. *point cloud*). Moderni laserski skeneri su u stanju prikupljati takve podatke s velikom brzinom i preciznošću, više od 1,000,000 točaka u sekundi s preciznošću boljom od 1 cm (Lemmens 2011).



Slika 2.1. Princip rada laserskog skenera (URL 2)

Laserski skeneri se primarno dijele na terestričke i one za lasersko snimanje iz zraka. Laserski skener korišten u konkretnom slučaju bio je terestrički laserski skener (TLS), stoga će se pobliže objasniti podjele te vrste laserskih skenera. S obzirom da ne postoji jedan univerzalni skener s kojim bi se mogla izvršiti sva mjerenja, tako postoje i različite vrste terestričkih laserskih skenera. Neki skeneri su bolji za zatvorene prostore i mjerenje srednjih dužina, neki su bolji za otvorene prostore i mogu mjeriti velike dužine, neki pak mogu mjeriti samo kratke dužine, neki imaju ograničen prozor snimanja, dok neki imaju veći prozor snimanja (Bojić i dr. 2017).

Također je vrlo teško napraviti jedinstvenu kategorizaciju TLS-a pa tako postoje razne podjele od kojih će se najviše pažnje obratiti na podjelu prema platformi snimanja, gdje imamo statične i dinamične TLS-e. Osim te podjele, još neki od načina grupiranja su (Lasić 2008):

1. Podjela prema načinu mjerenja udaljenosti na:
   a) pulsni,
   b) fazni,
   c) triangulacijski.
2. Podjela prema načinu snimanja ili vidnom polju na:
   a) skeneri – kamere,
   b) panoramski skeneri,
   c) hibridni skeneri.
3. Podjela prema načinu prikupljanja oblaka točaka na:
   a) TLS-i sa kompenzatorom sa mogućnošću dobivanja apsolutnih (georeferenciranih) podataka direktno na terenu.
   b) TLS-i bez kompenzatora kojima dobivamo relativni (lokalni) oblak točaka koji se također može georeferencirati, ali naknadno u računalnoj obradi.

Zanimljivo je napomenuti da se LiDAR ne koristi samo za prostorno skeniranje, već i za praćenje fizikalnih procesa u atmosferi, jer omogućuje vrlo precizno mjerenje brzine, smjera kretanja, te gustoće čestica u atmosferi. Ova tehnologija se obilježava kraticom DIAL (engl. *DIfferential Absorption LiDAR*).

2.1. Statični terestrički skeneri

Lasersko skeniranje se može podijeliti u dvije grupe: statično lasersko skeniranje i dinamično lasersko skeniranje. Kada je laserski skener fiksiran prilikom snimanja neke površine, onda se takav način laserskog skeniranja naziva statično lasersko skeniranje, a kada je laserski skener vezan za neku mobilnu platformu, naziva se dinamičnim laserskim skeniranjem. Ovakvi sustavi zahtijevaju i korištenje sustava za pozicioniranje (GPS) kako bi se odredio položaj laserskog skenera u prostoru prilikom snimanja. Primjeri dinamičnog laserskog skeniranja su: skeniranje iz aviona, iz automobila ili iz bespilotne letjelice (Rusov, 2014).

Statična metoda i statični laserski skeneri se koriste prvenstveno za precizna mjerenja gdje je zahtijevana točnost prioritet. Dobiveni 3D oblak točaka tada je vrlo gust pošto se u različitim vremenskim epohama i sa različitih stajališta snima uvijek isti objekt ili područje. Pogodni su za snimanje zatvorenih prostora gdje ne treba više stajališta, a njihova je primjena vrlo zastupljena pri monitoringu i inspekciji izgrađenih objekata (Slika 2.2.), npr. mostova, brana i prirodnih područja: klizišta i stijena. Također se primjenjuju za snimanje materijalne kulturne baštine te kod snimanja manjih objekata i artefakata. Nedostaci statičnih skenera su njihova težina, obrada velike količine podataka te problematika snimanja većih površina koje

zahtijevaju velik broj stajališta. Višestruka stajališta su otežavajuća okolnost pogotovo za naknadnu obradu gdje je, za pravilno spajanje više oblaka točaka u jedan jedinstveni, potrebna prilična vještina i iskustvo.

Slika 2.2 Statični laserski skener Faro Focus3D X 330 pri monitoringu mosta (URL 3)

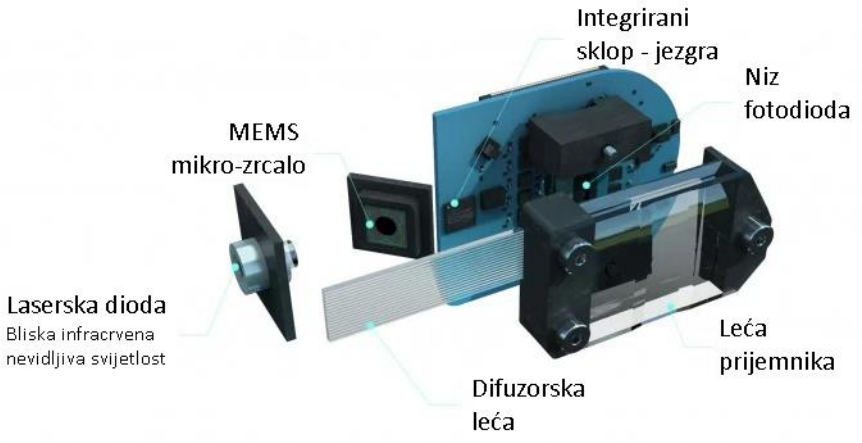
2.2. Dinamični laserski skeneri

Kao što je već i spomenuto, ova je vrsta skenera namijenjena za snimanje sa nekog pokretnog stajališta. To znači da TLS ne stoji nepomično na stativu, već je montiran na pokretnu platformu koju može nositi osoba koja opaža (Slika 2.3.), može biti pričvršćena na automobil, avion ili neko drugo prijevozno sredstvo. Ova vrsta skenera omogućuje brži obuhvat većeg prostora interesa, no ako se mjeritelj prebrzo kreće to može rezultirati manjom gustoćom prikupljenih točaka, a nagli pokreti grubim greškama. Preciznost ovih TLS-a u pravilu je manja nego kod statičnih skenera. Uzrok opadanja kvalitete rezultata je rjeđi oblak točaka te kumulativna pogreška koja nastaje prilikom kretanja (povećava se povećanjem duljine trajektorije). Pri snimanju otvorenih prostora također je vrlo teško izbjeći susret sa drugim ljudima ili nekim pokretnim objektima. Na kvalitetu utječu i svi ti pokretni objekti koji „zbunjuju“ uređaj koji ih pokušava uklopiti u prikaz. Uz sve ove degradirajuće faktore ova je metoda snimanja još uvijek zadovoljavajuća za namjene u kojima se koristi. Postoje i razni algoritmi i metode koji umanjuju ili uklanjaju pogreške. Kako bi se LiDAR-i točnije orijentirali u prostoru, često se kombiniraju sa GNSS uređajima i/ili sadrže IMU (engl. *Inertial Measurement Unit – Inercijalna mjerna jedinica*).

Slika 2.3. LiDAR Paracosm PX-80 realiziran na način da snima dok mjeritelj hoda (URL 4)

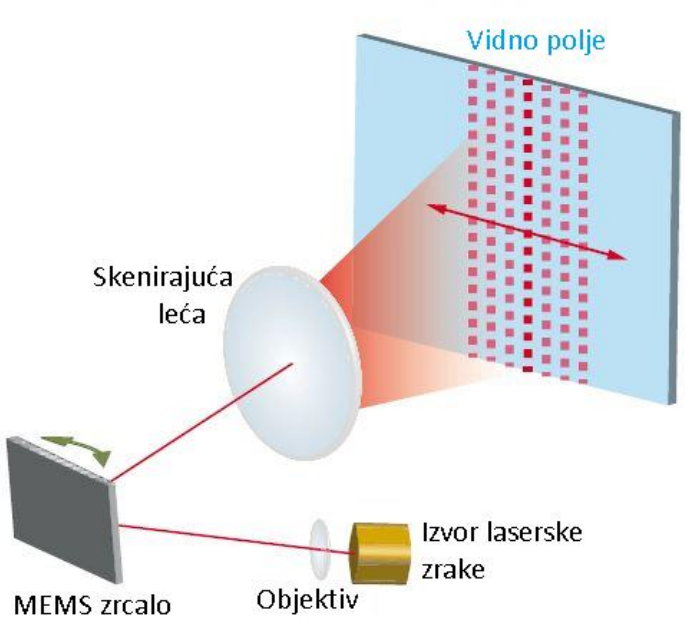
2.2.1. LiDAR čvrstog stanja (engl. *Solid State LiDAR*)

Sa mogućnošću pružanja velikog dometa i vrlo preciznih 3D mjerenja okoliša, LiDAR postaje bitan senzor u mnogim robotskim aplikacijama, kao što su autonomna vozila, dronovi, snimanje i kartiranje. Kako bi se omogućila masovna uporaba u tim područjima, nedavni razvoj LiDAR tehnologija usredotočen je na smanjenje troškova uređaja uz povećanje njegove pouzdanosti. U tom trendu, vrsta koja postaje sve interesantnija i konstantno se razvija je LiDAR čvrstog stanja. Dolazi u raznim implementacijama, kao što su LiDAR-i u cijelosti izgrađeni na silikonskom mikro-čipu (engl. *Micro Electromehanical System – MEMS*) (Slika 2.4.), oni koji rade na principu optičkog faznog niza (engl. *Optical Phase Array - OPA*), oni koji koriste elektro-optički senzor, tekuće kristale ili Risley-evu prizmu. S masovnom proizvodnjom, ovi LiDAR-i visokih performansi i ekstremno niske cijene drže potencijal za promicanje ili radikalnu promjenu robotske industrije (Lin i Zhang, 2019b).



Slika 2.4. Konstrukcija LiDAR-a čvrstog stanja (URL 5)

LiDAR čvrstog stanja je jedna od tehnologija koja bi mogla višestruko ubrzati i pojednostaviti geodetska mjerenja. Omogućuje detekciju, lociranje i mjerenje raznih objekata, što statičkih, što pokretnih, pa čak i tekućih tvari. Zbog svoje cijene ovi LiDAR-i su prihvatljivi širem spektru korisnika, dok se za skuplje i kompliciranije tradicionalne LiDAR-e to ne može reći. Ono što ovaj sustav čini puno jeftinijim od svog prethodnika su dvije stvari vezane uz samu proizvodnju. Kao što i sam naziv kaže, ovi LiDAR-i nemaju pokretnih dijelova. Leća je konstruirana na način da je smjer laserskog snopa promjenjiv (Slika 2.5.). Tako imamo uređaje kojima je horizontalno vidno polje ili FOV (engl. *Field Of View*) 10-ak stupnjeva, pa sve do 120°. Vertikalni doseg je po pravilu nešto manji i najčešće je ispod 40°. Odsutnost pokretnih mehaničkih dijelova ovoj vrsti LiDAR-a donosi veću kompaktnost, robusnost i mogućnost proizvodnje kućišta sve manjih dimenzija. Druga i najveća prednost ovog sustava je njegova cjelokupna konstrukcija na poluvodičkom silikonskom čipu. Takva konstrukcija je također utjecala na višestruko smanjenje veličine mjernog uređaja, a nastavno na to, i smanjenje težine i jačine napajanja potrebnog za rad samog uređaja. Nakon svega navedenog dolazi se do zaključka da i uz višestruko smanjenje cijene ovi LiDAR-i ne zaostaju značajno za svojim prethodnicima, a zbog jednostavnosti građe čak su i pouzdaniji pri korištenju na duži vremenski period (URL 6).



Slika 2.5. Princip rada LiDAR-a čvrstog stanja baziranog na MEMS čipu (URL 7)

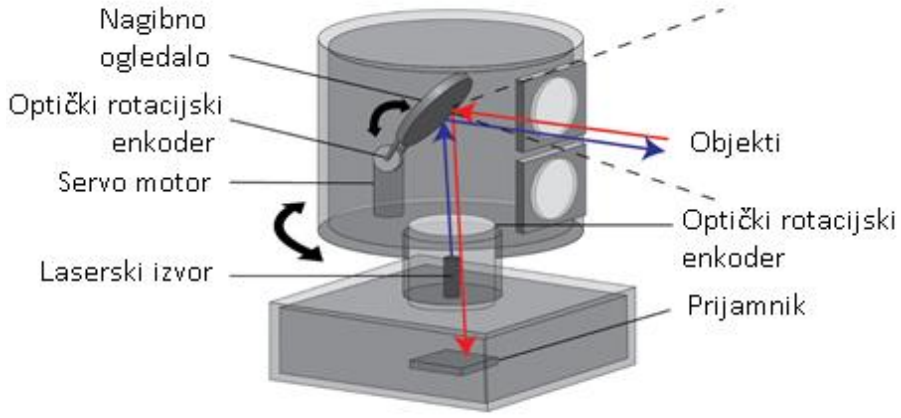
Prvenstvena svrha za masovnu proizvodnju ovih LiDAR-a je automobilistička industrija. Sve se više teži ka vozilima koja se mogu kretati autonomno bez potrebe za vozačem. Ta bi tehnologija uvelike olakšala vožnju ljudima, a i transport robe bilo bi moguće automatizirati. Također bi se svakim novim vozilom, koje pomoću senzora može prepoznati, a onda i reagirati na sve predvidive i nepredvidive situacije, povećala sigurnost prometa i smanjile ljudske žrtve u prometu (kojih u svijetu ima oko 1,35 milijuna godišnje). Uz sve to, takav novi način upravljanja donosi i financijsku dobit (URL 8).

No, nije samo automobilska industrija ona koja može iskoristiti LiDAR-e čvrstog stanja. Zbog visokih frekvencija prikupljanja točaka i sve većeg daljinskog dosega ovi LiDAR-i mogu višestruko ubrzati proces snimanja i kartiranja okoliša. U relativno kratkom vremenu moguće je dobiti vrlo gust oblak točaka koji u trodimenzionalnom smislu može predočiti pojedine objekte ili čak veća otvorena područja. Kasnije se taj model može koristiti kao podloga nekim narednim radovima ili kao model terena u različite svrhe. Ova metoda izmjere također je pogodna za korištenje i u zatvorenim prostorima.

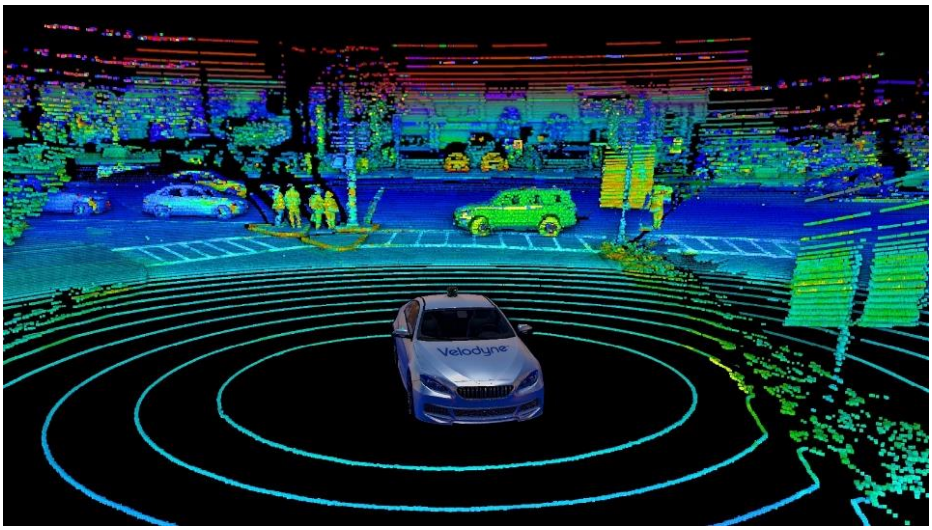
Posljedično na sve veću potražnju, dolazi i do sve veće ponude i sve boljih performansi LiDAR-a čvrstog stanja. Neke od vodećih tvrtki za proizvodnju takvih LiDAR-a su Livox, Velodyne, Ouster, ibeoNEXT itd. koje sve redom pokušavaju zadovoljiti kriterije i zahtjeve tržišta. Svaka tvrtka uz proizvodnju čim boljeg uređaja veliku pažnju pridaje i razvoju vlastitih računalnih sustava, te korisničkoj podršci. Na taj način je korištenje uređaja lako savladati, a sam uređaj se može koristiti u najrazličitije svrhe i implementirati u različite sustave.

2.2.2. Mehanički LiDAR

Mehanički LiDAR se od LiDAR-a čvrstog stanja razlikuje, kako i samo ime kaže, po mehaničkim dijelovima u svojoj konstrukciji (Slika 2.6.). Ova vrsta LiDAR-a ima pokretne dijelove koji služe u prvom redu za povećanje vidnog polja koji za ovu vrstu LiDAR-a horizontalno doseže cijeli krug, tj. 360° (Slika 2.7.). Pokretni dijelovi omogućuju najveću prednost ove vrste LiDAR-a, što je vidno polje cijelog kruga, no s druge strane oni također prouzrokuju njihove mane u vidu cijene izrade, osjetljivosti na vibracije, manje robusnosti i veće težine koje je teško svesti na veličinu LiDAR-a čvrstog stanja. Ovi se LiDAR-i najčešće postavljaju na krov automobila kako bi postigli svoj puni potencijal. Već u slučaju da ih se postavi na prednji dio automobila, gubi se pola vidnog polja čime se dolazi do zaključka da je isplativije i sigurnije rješenje kombiniranje više malih, jeftinih i robusnih LiDAR-a čvrstog stanja (URL 8; URL 9).

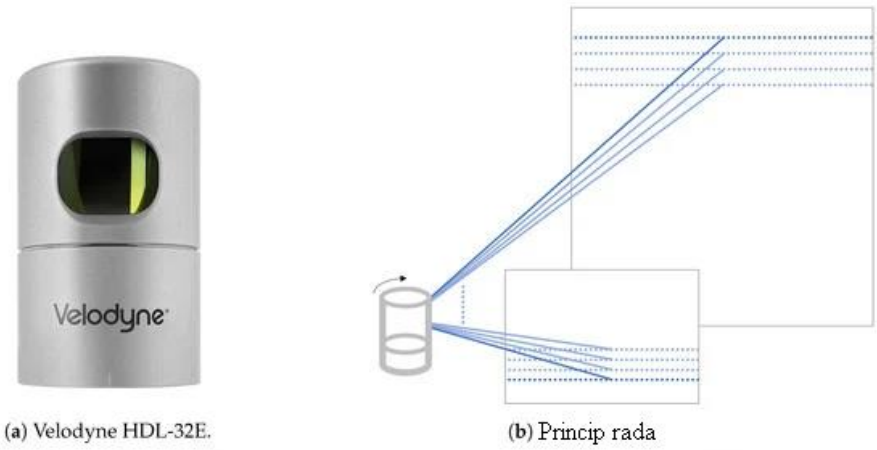


Slika 2.6. Konstrukcija rotirajućeg LiDAR-a (URL 9)



Slika 2.7. Prikaz vidnog polja rotirajućeg LiDAR-a postavljenog na krovu automobila (URL 10)

Uz horizontalni FOV od 360°, ova vrsta LiDAR-a najčešće pokriva između 30° i 40° vertikalnog FOV-a. Ovisno o vertikalnom vidnom polju i broju kanala odnosno laserskih prstenova dobiva se rezolucija uređaja i pokrivenost snimanog područja. Broj kanala kreće se od 8 do 128, a povećanje brzine kretanja platforme zahtjeva i povećanje kanala kako se ne bi izostavili bitni detalji u okolišu. Neke od najzastupljenijih tvrtki na tržištu koje proizvode rotacijske LiDAR-e su Velodyne, Hesai, Ouster i RoboSense, a na Slici 2.8. je prikazan Velodyne-ov 32-kanalni model HDL-32E.



Slika 2.8. Velodyne HDL-32E mehanički LiDAR i njegov princip rada (URL 11)

2.2.3. Livox Mid-100

Između sve veće ponude LiDAR-a čvrstog stanja trebalo se odlučiti za jedan koji će se koristiti u konkretnoj implementaciji. Odabrana je kompanija Livox Technlogy Company Limited i njihov Mid-100 LiDAR. Ta je tvrtka osnovana 2016. godine u svrhu razvijanja LiDAR sustava i prilagođavanje istih kupcima iz raznih područja. Uz sam „hardver“ nude veliku pomoć i putem online zajednice gdje se nudi velik broj softverskih rješenja za različite uporabe i ugradnju u projekte različite namjene. Kao i drugi srodni uređaji, LiDAR-i ove tvrtke se široko primjenjuju u industrijama poput autonomne vožnje, robota, 3D kartiranja, pametnih gradova, sigurnosti i sličnom.

Ključna prednost zbog koje je odlučeno korištenje uređaja ove tvrtke je upravo spomenuta podrška kojoj se može besplatno pristupiti na Wiki stranicama kompanije i GitHub-u. Za pokretanje samog sustava u najvećoj su se mjeri pratile smjernice ispisane od strane kolega sa sveučilišta u Hong Kongu koji su se u svom projektu bavili implementacijama Livox Mid-40 i Livox Mid-100 LiDAR-a i kamere u SLAM sustav za 3D kartiranje okoliša. Prvotna ideja bila je nabavka i korištenje Livox Horizon LiDAR-a za ovaj diplomski rad pošto taj model sadrži i IMU jedinicu i manji je od korištenog modela. Međutim taj model nije bio dostupan kod proizvođača tako da se drugi model, Livox Mid-100 (Slika 2.9.), posudio od Prometnog fakulteta koji ga ima u posjedu.

Slika 2.9. Livox Mid-100 LiDAR korišten u ovom diplomskom radu (URL 12)

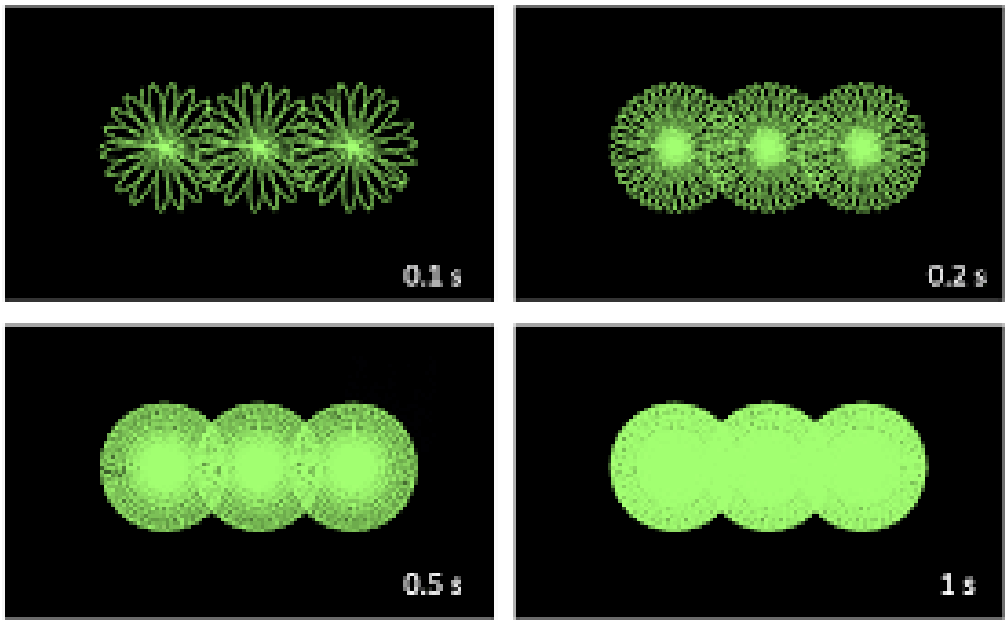
Livox-ov model Mid-100 sastoji se od 3 LiDAR jedinice što povoljno utječe na njegovo vidno polje. Svaka jedinica prikuplja 100 000 točaka u sekundi što ukupno daje frekvenciju od 300 000 točaka u sekundi. Domet mu je 260 metara, a proizvođač garantira točnost od 2cm. Sve specifikacije uređaja prikazane su u Tablici 2.1.

Tablica 2.1 Specifikacije LiDAR-a Livox Mid-100 (URL 13)

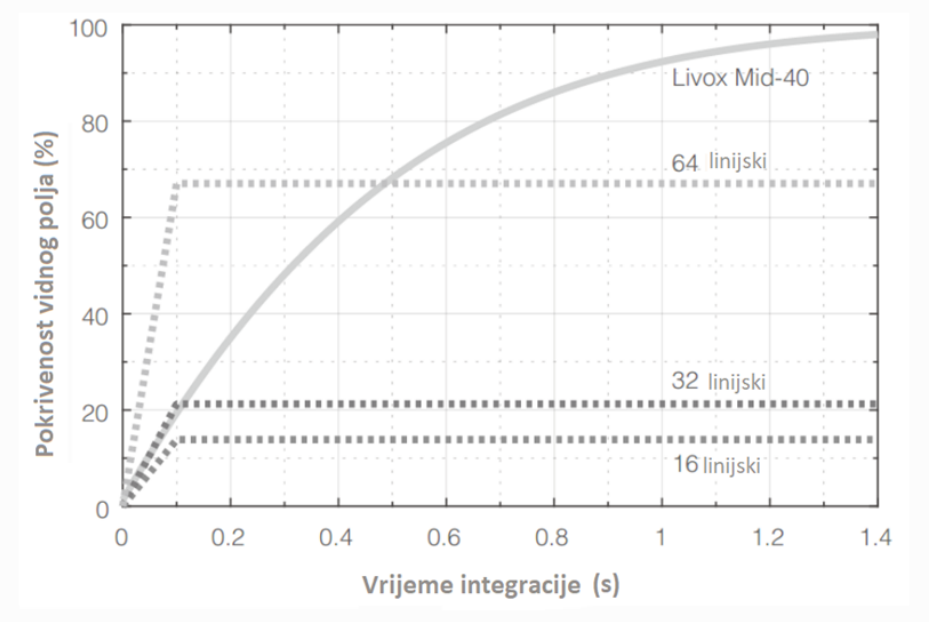
Specifikacija	Vrijednost
Model	Livox Mid-100 LiDAR senzor
Valna duljina lasera	905 nm
Sigurnost lasera	Klasa 1 (IEC60825-1)
Domet detekcije (@ 100 klx)	90 m pri 10% refleksije 130 m pri 20% refleksije 260 m pri 80% refleksije
Vidno polje	98.4° (Horizontalno) x 38.4° (Vertikalno)
Preciznost dometa (1σ @ 20 m)	2 cm
Kutna preciznost	<0.1°
Divergencija snopa	0.28° (Vertikalno) x 0.03° (Horizontalno)
Brzina mjerenja snopa	300 000 točaka/s

Pokrivenost vidnog polja	20% pri 0.1 s, 68% pri 0.5 s, 93% pri 1 s
Radna temperatura	-20°C do 65°C (-4°F do 149°F)
IP ocjena (rangiranje/rating)	IP67
Napajanje	Tipično 30 W
Napon	10 ~ 16 V DC (Tipičnp 12 V DC)
Sučelje	Ethernet
Sinkronizacija podataka	IEEE 1588-2008 (PTPv2), PPS *(Pulse Per Second*)
Uređaj	
Dimenzije	142 x 70 x 230 mm
Težina	≈ 2200 g
Kašnjenje podataka/latencija	2 ms

Velika pokrivenost opažanog područja u kratkom vremenskom intervalu postignuta je novom tehnologijom snimanja koju su u Livox-u nazvali nerepetitivnim uzorkom skeniranja (engl. *Non repetitive scanning pattern*). Laserska zraka tako ne popunjava vidno polje linijskim uzorkom već se kreće unutar kruga i pritom ne ponavlja prelazak preko već snimljenog područja. Ova metoda omogućuje stopostotnu pokrivenost vidnog polja već nakon nešto više od jedne sekunde snimanja. Uzorak snimanja može se vidjeti na Slici 2.10., dok se usporedba sa linijskim načinima snimanja može vidjeti na Slici 2.11. Na grafu je prikazan podatak za Livox Mid-40 model, no taj se podatak može primijeniti i na Mid-100 pošto radi po istom principu, jedino je domet u horizontalnom smislu utrostručen pošto Mid-100 model sadrži tri jedinice Mid-40. Lako se može uočiti zamjetna razlika između ove i prijašnjih metoda, te se vidi kako je pokrivenost vidnog polja već nakon malo više od pola sekunde visokih 80%, dok se 100% postiže već za 1,4 sekunde.

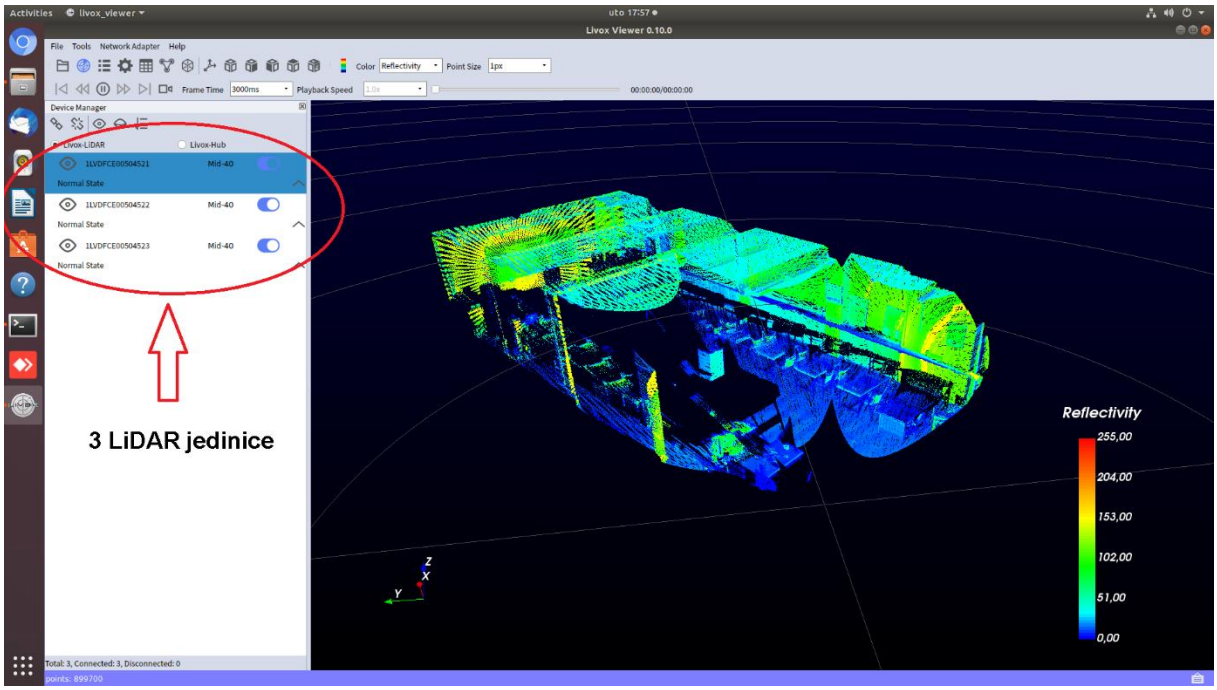


Slika 2.10. Nerepetitivni uzorak skeniranja (URL 14)



Slika 2.11. Usporedba nerepetitivnog skeniranja sa 16, 32 i 64 linijskim skeniranjem (URL 14)

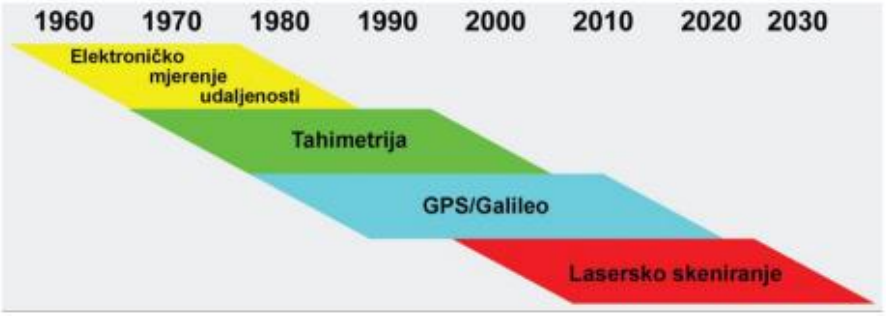
Tvrtka Livox nudi i vlastiti softver LivoxViewer za vizualizaciju u realnom vremenu kako bi se u svakom trenutku moglo provjeriti koje područje je zahvaćeno laserskim snopom (Slika 2.12.). Na slici je vidljivo kako se koriste istovremeno 3 senzora i intenzitet refleksije snimanih površina.



Slika 2.12. Prikaz trenutnog vidnog polja u softveru za vizualizaciju LivoxViewer-u

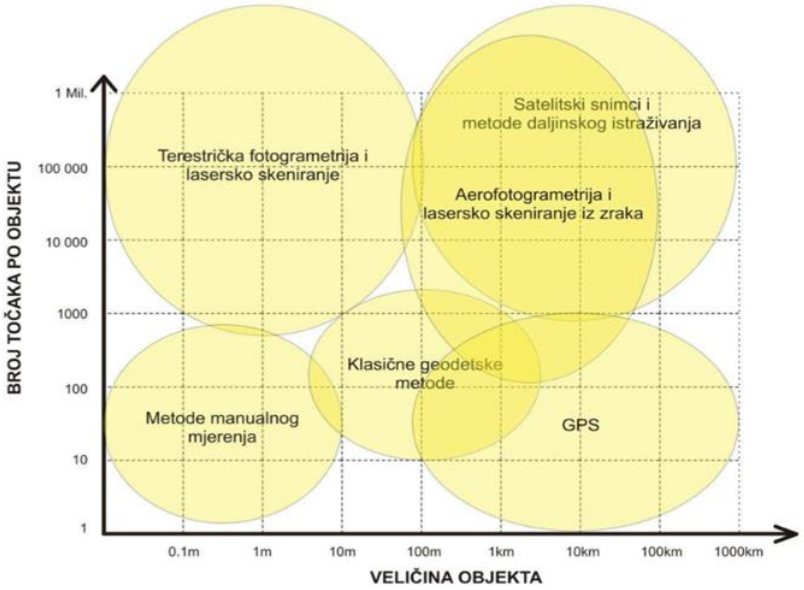
3. TERESTRIČKO LASERSKO SKENIRANJE

Terestričko lasersko skeniranje (engl. *Terrestrial laser scaning – TLS*) je relativno nova i vrlo učinkovita metoda pridobivanja detaljnih digitalnih snimaka velikih objekata, pa čak i cijelih područja manjeg i srednjeg protezanja (Miler i dr. 2007). S početkom 21. stoljeća terestričko lasersko skeniranje je preraslo iz polja istraživanja u tehnologiju spremnu za terensku primjenu koju nude brojne geoinformacijske firme diljem svijeta (Slika 3.1.). Tehnologija se primarno koristi za brzo prikupljanje trodimenzionalnih podataka objekata od interesa poput kulturnih znamenitosti, mostova, tvornica, tunela i brojnih drugih (Lemmens, 2011).



Slika 3.1. Povijesni razvoj tehnologija za snimanje prostora (Miler i dr., 2007)

Zbog mnogih prednosti terestričkog laserskog skeniranja, poput mjerenja bez kontakta s objektom, brzog dobivanja podataka, detaljnosti izmjere, omjera uloženog truda i dobivenih podataka, njegova upotreba je sve veća (Slika 3.2.). Primjenu terestričkih laserskih skenera susrećemo u raznim područjima poput mjerenja građevinskih objekata, deformacija na branama, topografskoj i industrijskoj izmjeri, arheološkim mjerenjima kulturnih objekata pa sve do suvremenih 3D katastara (Matijević i Roić, 2002).



Slika 3.2 Usporedba geodetskih metoda izmjere (Böhler i Heinz, 1999)

Princip rada je vrlo sličan današnjim klasičnim geodetskim instrumentima s laserom koji su u upotrebi već godinama. Laserska zraka odaslana iz mjernog instrumenta se reflektira od objekta snimanja i vraća natrag do mjernog instrumenta. Kombinacija izmjerene udaljenosti i kuta s određenog stajališta daje koordinate tražene točke u 3D prostoru (Miler i dr., 2007). Postupak se može podijeliti u četiri koraka (Berta, 2017):

1. Senzor odašilje lasersku zraku uz precizno bilježenje vremena i kuta pod kojim je ona odaslana.
2. Refleksija laserske zrake ili njenog dijela se detektira, također uz precizno bilježenje vremena. U slučaju kada zraka prolazi kroz vegetaciju, detektiraju se sve naknadne refleksije dok se kumulativno ne reflektira cijela zraka.
3. Korištenjem vremenske razlike između odašiljanja i refleksije laserske zrake, uz poznatu konstantnu brzinu svjetlosti, računa se udaljenost između senzora i objekta.
4. Na temelju izračunate udaljenosti između senzora i objekta, kuta pod kojim je laserska zraka odaslana i preciznog položaja senzora računaju se prostorne koordinate točaka reflektirajuće površine.

Sve veći interes za LiDAR-ima nikako ne čudi jer oni imaju mnoge prednosti u usporedbi sa drugim geodetskim metodama izmjere (Slika 3.3.). LiDAR u usporedbi sa totalnom stanicom u puno kraćem vremenskom razdoblju može prikupiti puno veći broj točaka. Uspoređujući ovu metodu sa fotogrametrijskom također se nailazi na jednu veliku prednost LiDAR-a, a to je neovisnost o danjem svijetlu tokom snimanja i dobivanje direktno mjerene udaljenosti. Iako fotogrametrija kombiniranjem više fotografskih snimaka može dati vjeran 3D model nekog

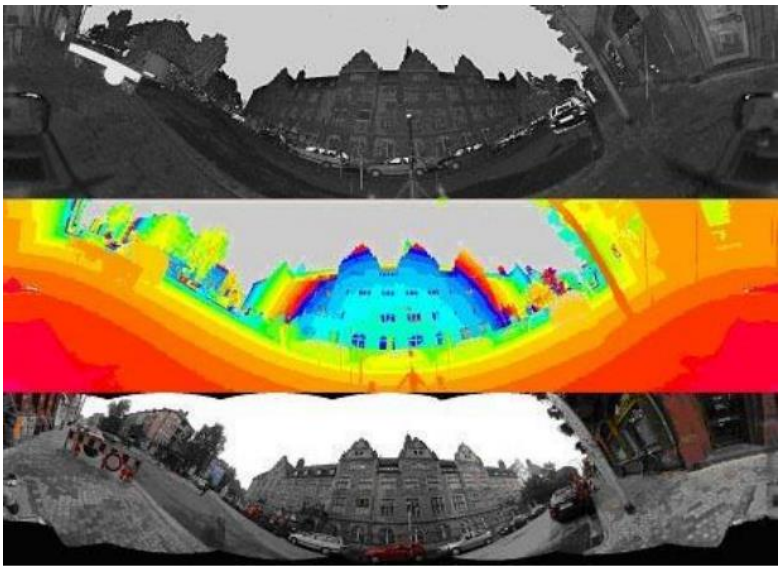
objekta, njoj je za to potrebno dobro osvjetljenje svih dijelova tog objekta, u protivnom dolazi do šuma na sjenovitim mjestima. Pošto LiDAR funkcionira pomoću aktivnog snopa laserskih zraka, njemu nije potrebno svijetlo, pa se tako njime može snimati i noću. Uz snimanje noću moguće je snimati i loše osvjetljene zatvorene prostore, pa čak i primjerice mračne špilje kod kojih je izmjera nekim drugim metodama puno kompliciranija.

Slika 3.3. Usporedba različitih karakteristika geodetskih metoda izmjere (Zogg, 2008)

3.1. Statično terestričko lasersko skeniranje

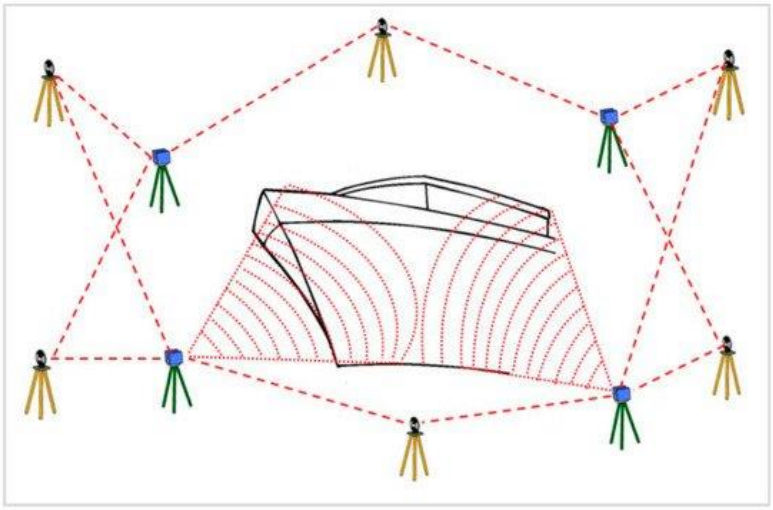
Statično terestričko lasersko skeniranje (STLS) je metoda laserskog skeniranja pri čemu je laserski skener fiksan (nepomičan) prilikom snimanja neke površine. Posljednjih godina, statički TLS uspostavljen je kao standardni postupak koji može zadovoljiti uvjete brzog snimanja 3D objekata. Princip rada statičkog skeniranja temelji se na ponovljenom mjerenju raspona nagiba, određenog pomoću skenera koji mjeri udaljenosti u vodoravnoj i okomitoj ravnini. Rezultat procesa su sferne polarne koordinate točaka u vidnom polju skenera u lokalnom koordinatnom sustavu (Gikas, 2012).

Instrumenti za STLS uglavnom koriste pulsnu, faznu ili triangulacijsku metodu mjerenja udaljenosti. Osnovni princip je sličan principu rada totalne stanice, a za izračun duljine mjeri se vrijeme potrebno da se odaslana zraka odbije i vrati do uređaja. Međutim, postoje značajne razlike u valnim duljinama lasera, količini i brzini prikupljanja točaka, pripremi terena, obradi podataka, izvorima pogrešaka, itd. Laserskim skeniranjem se dobiva velika količina neobrađenih podataka u obliku 3D oblaka točaka. Kad su kontrolne točke georeferencirane u poznatom koordinatnom sustavu, tad se čitavi oblak točka može orijentirati u istom sustavu. Sve točke unutar oblaka točaka imaju X, Y i Z koordinate i podatak o intenzitetu, te se tako dobiva XYZI format točaka. Intenzitet predstavlja amplitudu povratnog laserskog signala i ovisi o apsorpciji i transparentnosti materijala, te je različit za svaki materijal. Točke mogu biti dodatno određene i bojom, tj. mogu se definirati i RGB komponentnom u slučaju kada skeneri imaju u sebi integriranu i kalibriranu digitalnu kameru (Slika 3.4.).



Slika 3.4. Prikaz intenziteta, udaljenosti od objekta i pomoću RGB palete boja (Kordić, 2014)

Za dugačke, složenije objekte potrebno je nekoliko stajališta u klasičnom pristupu statičkog skeniranja. 3D oblaci točaka pojedinačnih stajališta moraju se referencirati ili georeferencirati kako bi se svi 3D oblaci točaka prebacili u zajednički koordinatni sustav s poznatim geodetskim datumom. Povećanjem broja stajališta, taj postupak postaje dugotrajan i neispravan, pa se ipak preporučuje korištenje dinamičkog TLS-a. Kod dinamičkog TLS-a nedostatak čini korištenje dodatnih senzora koji povećavaju troškove sustava. Samo povezivanje može se postići postavljanjem točaka (prirodnih ili umjetnih), koje moraju biti prisutne u oba susjedna 3D oblaka točaka. Na Slici 3.5. dan je primjer mjerenja objekta s nekoliko stajališta, pri čemu su zeleno označena stajališta statičkog TLS-a, a žuto signalizirane mete (Stenz i dr., 2020).



Slika 3.5. Princip mjerenja objekta statičkom laserskom metodom (Stenz i dr., 2020)

3.2. Mobilno lasersko skeniranje

Mobilno lasersko skeniranje (engl. *Mobile Laser Scanning – MLS*) predstavlja najsuvremeniju metodu mjerenja koja je doživjela veliki razvoj u posljednjem desetljeću. U slučaju mobilnog laserskog skeniranja uređaj mijenja svoj položaj tijekom snimanja podataka. Iz mjerenja udaljenosti i kuta te kretanja skenera nastaje 3D oblak točaka. Dinamični skeneri mogu se postaviti na različite platforme poput vagona, automobila, kamiona, broda i slično. Ovaj način mjerenja može s visokom točnošću i gustoćom snimiti i do tisuću točaka po kvadratnom metru (Lin i dr., 2014).

Ova vrsta laserskih skenera nerijetko je uparena sa drugim tipovima opreme, koristeći GNSS prijemnike i IMU. U tom slučaju pozicija i orijentacija skenera se određuje kombinacijom podataka s GNSS-a, IMU-a i preciznog odometra. IMU koristi magnetometar, akcelerometar i žiroskop kako bi se kontinuirano računala pozicija, orijentacija, brzina i smjer kretanja vozila bez uzimanja u obzir vanjskih referenci. IMU se koristi na vozilima kao što su brodovi, avioni, podmornice, navođeni projektili i terestričkim MLS sustavima. Unutar terestričkih MLS sustava IMU se koristi kako bi se odredila promjena XYZ pozicije i orijentacije senzora između GNSS mjerenja i tijekom perioda kada je GNSS nedostupan. Zbog svoje pokretnosti MLS nudi nekoliko prednosti u odnosu na tradicionalne metode za mjerenja koridora, uključujući veliku brzinu sakupljanja podataka (smanjeno vrijeme i troškovi), veliku gustoću prikupljenih podataka u vidu 3D oblaka tačaka, tako da ni detalji poput primjerice električnih vodova ili protezanja krošanja drveća nisu izostavljeni. Usporedno sa statičnim skeniranjem, gdje za snimanje određenog područja postoji limitiran broj stajališta, ova metoda je pogodnija za snimanje velikih prirodnih površina, te za snimanja više objekata koji se tada snimaju sa svih strana i lakše se izbjegavaju prepreke (npr. snimanje između zgrada). Gustoća oblaka točaka je funkcija frekvencije prikupljanja točaka i brzine vozila. Napredni terestrički MLS sustavi omogućavaju korisniku visoko precizne podatke u kratkom vremenskom roku.

3.2.1. Simultana lokalizacija i kartiranje – SLAM

Zbog nedostataka kod klasičnih skenera poput vremenske ograničenosti snimanja, obrade podataka, težine uređaja, zahtjevnosti snimanja objekata poput špilja, šuma i slično, javlja se potreba za mobilnim sustavima koji su praktičniji i jednostavniji za uporabu. SLAM (engl. *Simultaneous Localization and Mapping*) je prvobitno nastao u robotici za potrebe navigacije autonomnih vozila u nepoznatim prostorima bez sudaranja u zidove, ljude, ili neke druge prepreke (URL 15). SLAM spada među dinamične metode laserskog skeniranja. To je proces pomoću kojeg robot može izraditi kartu okruženja i istodobno koristiti istu kartu pri određivanju svoje lokacije (relativne pozicije i orijentacije u okolini), a implementiran je u niz različitih područja, od unutarnjih do vanjskih robota, terestričkih, podvodnih i zračnih sustava. SLAM uređaji koriste podatke senzora kako bi odredili svoju trenutnu poziciju i kreirali sliku prostora temeljem oblaka točaka (Durrant-Whyte i Bailey, 2006).

U autonomnim vozilima i ostalim autonomnim robotima koji imaju svijest o tome gdje su i koji je najbolji put do mjesta gdje idu SLAM je ključna komponenta. Stvaranjem vlastitih karata SLAM omogućuje brži, autonomniji i prilagodljiviji odgovor nego što ga daju unaprijed programirane rute (URL 16).

Najjednostavniji oblik istovremenog lociranja i kartiranja gdje je primarni senzor opažanja kamera ili neki drugi slikovni senzor naziva se vizualni SLAM. Iako najjednostavniji oblik SLAM-a, pruža bogat izvor informacija o strukturi. Može koristiti jednostavne kamere (širokokutne, „riblje oko“ i sferne kamere), složene kamere (stereo i više kamera) i RGB-D kamere. Vizualni se SLAM može implementirati po relativno niskoj cijeni s jeftinim kamerama (URL 16). Nedavno je korištenje vizualnih senzora postalo važan aspekt istraživanja SLAM-a, dijelom zato što slika pruža bogat izvor informacija o strukturi. Veliki broj istraživanja provođen na vizualnom SLAM-u koristio je stereokamere ili kamere zajedno s drugim senzorima (poput akcelerometara ili GPS-a), no od 2001. godine brojni su radovi pokazali kako se SLAM može uspješno provesti upotrebom samo jedne kamere (poznate kao monokularni vizualni SLAM), na primjer, seminarski rad Andrewa Davisona na Sveučilištu u Oxfordu. (URL 17). LiDAR SLAM je metoda koja primarno koristi laser. Laseri, u usporedbi s kamerom, su puno pouzdaniji i trebaju manje resursa za obradu podataka, te se koriste u pokretnim objektima, kao što su auti bez vozača i dronovi. Izlazni podaci laserskih skenera su uglavnom oblaci točaka (URL 16).

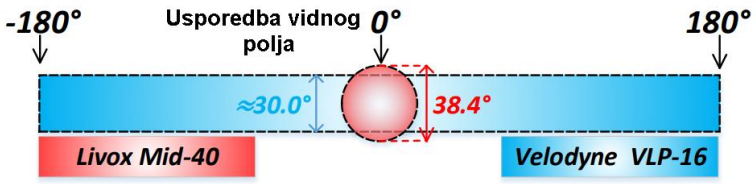
Algoritam SLAM-a radi na principu iterativnog postupka pri određivanju trajektorije kretanja i za svaku točku trajektorije stvara 3D oblak točaka. U svakom koraku trajektorije, vremenski prozor obrade poveća se za manji dio duljine te se algoritmom uklapanja oblaka točaka na temelju istih ploha, novi oblak točaka uklapa na postojeći dio oblaka. Budući da su točnost i preciznost senzora od velike važnosti, većina predloženih sustava uključuje primjenu skupih laserskih senzora (Ozisik i Yavuz, 2016).

Tijekom samog procesa skeniranja, na završetku je bitno uređaj odložiti na istu horizontalnu plohu s koje je skeniranje započeto – zatvaranje petlje (engl. *Loop Closure*) – postiže se veća točnost budući da povećanjem duljine trajektorije opada točnost inercijalnih senzora i SLAM algoritma (Nocerino i dr. 2017). Zatvaranje petlje predstavlja bitan dio SLAM sustava za procjenu dugoročnog akumuliranog pomaka uzrokovanog lokalnim podudaranjem značajki. Za razliku od vizualnog SLAM-a, istraživanja o zatvaranju petlje koristeći laserske sustave su rijetka te je vrlo teško naći rješenja otvorenog koda koja bi mogla pomoći riješiti ovaj problem. Kod laserskog SLAM-a veliki problem predstavlja otkrivanje petlje za podatke oblaka točaka. To je uglavnom jer su LiDAR senzori, u usporedbi s kamerama prilično skupi, što sprječava njihovu širu uporabu, a i u robotskoj navigaciji često LiDAR nije prvi izbor. Nadalje, velik izazov predstavlja i prepoznavanje mjesta u oblaku točaka, gdje su nam dostupne samo informacije o geometrijskim oblicima u 3D prostoru, za razliku od 2D slika koje sadrže bogate informacije poput boja i struktura (Lin i Zhang, 2019a).

Za ispravan rad SLAM-a bitna je i geometrija oblika u okolini. Idealna bi okolina bila s različitim oblicima i razvedenim plohama. Prostor, odnosno predmet skeniranja bi trebao imati dovoljan broj karakterističnih značajki na temelju kojih će se vršiti preklapanje (Nocerino i dr. 2017). Izazov za robusna SLAM rješenja predstavljaju okruženja koja uzrokuju nastanak jakih i učestalih šumova. Primjena SLAM-a uključuje i neistražena područja kao što su područja prirodnih katastrofa, podmorje, i mnoge druge lokacije.

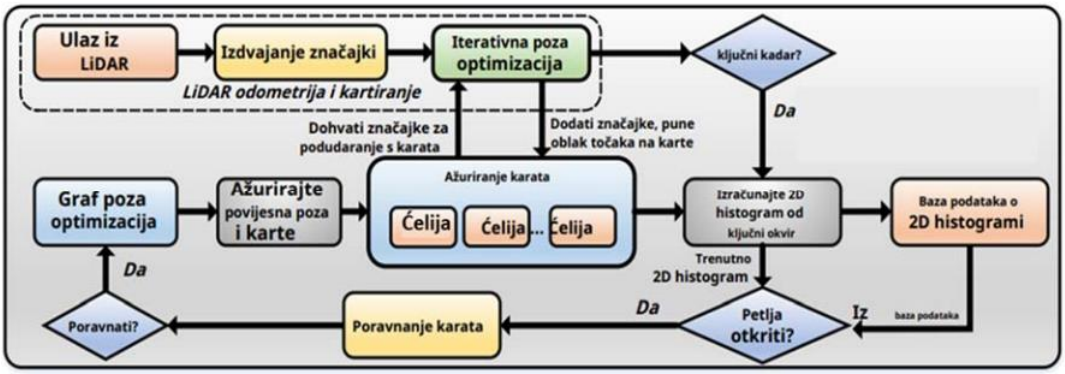
3.2.2. LOAM Livox

Uz sve prednosti Livox LiDAR senzora, sama tehnologija još uvijek ima mane koje degradiraju dobivene podatke i samu primjenu. Neke od njih su relativno malo vidno polje LiDAR-a čvrstog stanja i konkretno Livox Mid-100 u odnosu na rotirajuće LiDAR-e (rješava se kombiniranjem više jedinica) (Slika 3.6) , nepravilan i nerepetitivni uzorak skeniranja, te zamućenje uzrokovano kretanjem i rotacijom uređaja. Kako bi se riješili navedeni problemi razvijen je softverski paket LOAM Livox (engl. *LiDAR Odometry And Mapping*).



Slika 3.6. Usporedba vidnog polja Livox Mid-40 sa rotirajućim Velodyne VLP-16 (Lin i Zhang, 2019a)

Uobičajen način rješavanja problema pozicioniranja je korištenje dodatnih senzora poput GNS-a i INS-a (engl. *Inertial Navigation System*) za registriranje laserskih točaka u određeni koordinatni sustav. Drugi način rješavanja je korištenje metoda poput odometrije ili vizualnih sustava za mjerenje kilometraže. Odometrija može rezultirati prevelikim odmakom od prave vrijednosti pa se pomoću zatvaranja petlje pokušava umanjiti pogreška. LOAM sustav zatvaranja petlje za LiDAR odometriju je sustav koji se sastoji od brzog otkrivanja petlje, poravnavanja karata i optimizacije grafa. Metoda je brza, otporna na rotaciju te proizvodi pouzdano i točno otkrivanje petlje, što dovodi do cjelovitog i praktičnog sustava za upotrebu. Na Slici 3.7. prikazan je tijek rada sustava, u kojem se svaki novi okvir (skeniranje) LiDAR-a registrira na globalnoj karti pomoću LOAM algoritma (Lin i Zhang, 2019a).



Slika 3.7. Tijek rada LOAM sustava (Lin i Zhang, 2019a)

LOAM Livox je robustan paket za odometriju i kartiranje u realnom vremenu s malim pomakom namjenjen Livox LiDAR-ima. Ovaj paket rješava mnoga ključna pitanja: izdvajanje i odabir bitnih značajki u vrlo ograničenom vidnom polju, robusno odbacivanje grubih pogrešaka, filtriranje pokretnih objekata i kompenzaciju izobličenja zbog kretanja. Sve ove zadaće rješavaju se bez drugih senzora poput IMU-a, GNSS-a i kamera. Algoritam računa lokaciju LiDAR-a u stvarnom vremenu, a pritom registrirajući oblak točaka (Lin i Zhang, 2019b).

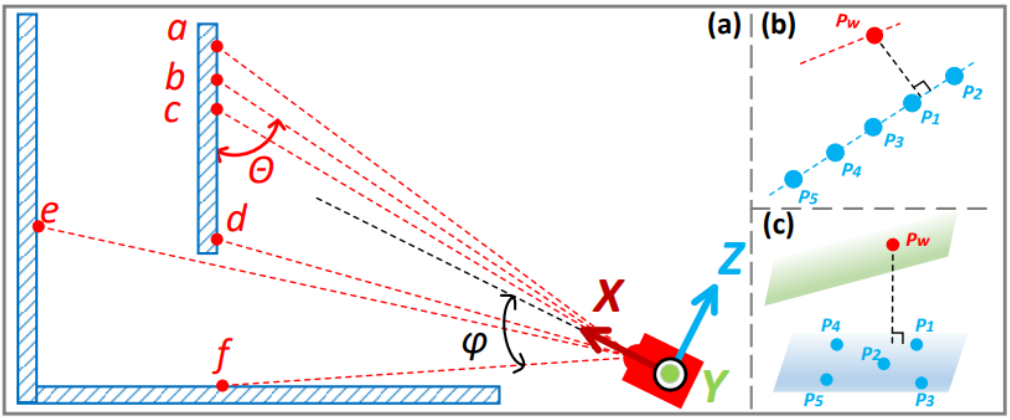
Uspješno je primijenjen u području autoindustrije, robotike, autonomnih dronova i sličnim područjima. U slučaju kada je LiDAR statičan i jedino se okreće laserska zraka, registracija laserskih točaka je jednostavna, ipak, ako se sama LiDAR jedinica okreće u raznim smjerovima potrebno je poznavati položaj LiDAR-a tijekom cijelog skeniranja (Lin i Zhang, 2019b).

Ovaj algoritam postiže visoku razinu točnosti u kartiranju i lokalizaciji, kao i u skeniranju. Neki od doprinosa metode su:

1.) Razvijanje kompletnog softverskog paketa za odometriju i kartiranje LiDAR-om koji je otvoren, besplatan i dostupan svima u svrhu doprinosa zajednici.
2.) Povećanje točnosti i robusnosti algoritma načinom 'front-end' procesiranja.
3.) Jednostavna ali efikasna metoda kompenzacije kretanja, bolja postepena obrada i linearna interpolacija podataka.

Uspoređujući karakteristike svake snimljene točke izvršava se odabir „dobrih“ i odbacivanje „loših“ točaka, te ekstrakcija značajki „dobrih“ točaka. Odabir korisnih točaka vrši se usporedbom daljine, kuta i intenziteta odbijene zrake pojedine točke sa istim tim karakteristikama susjednih točaka (Slika 3.8. (a)). Algoritam tako, primjerice, odbacuje sve točke sa ruba vidnog polja ($\varphi(P) \geq 17°$ za Livox Mid-40) – na slici točka P_f , točke sa prevelikim ili premalim intenzitetom (točka je ili predaleko ili površina ima premalu reflektivnost), točke sa kutom odbijanja blizu 0° ili 180° - na slici točka P_f, te točke skrivene iza objekata – na slici točka P_e. Nakon odabira „dobrih“ točaka dodjeljuju se karakteristike pojedinim točkama, pa se tako odvajaju točke na plohi od onih koje označavaju rub neke plohe (Slika 3.8. (c)). Da bi se

ublažile nepravilnosti nastale manjkom podataka zbog ograničenog vidnog polja, uvodi se reflektivnost kao mjerenje četvrte dimenzije svake točke. Ako dvije susjedne točke imaju vidno različitu reflektivnost dodjeljuje im se značajka rubnih točaka dvaju ploha ili različitih materijala (npr. razlikovanje zatvorenih vrata ili prozora od zida) (Lin i Zhang, 2019b).



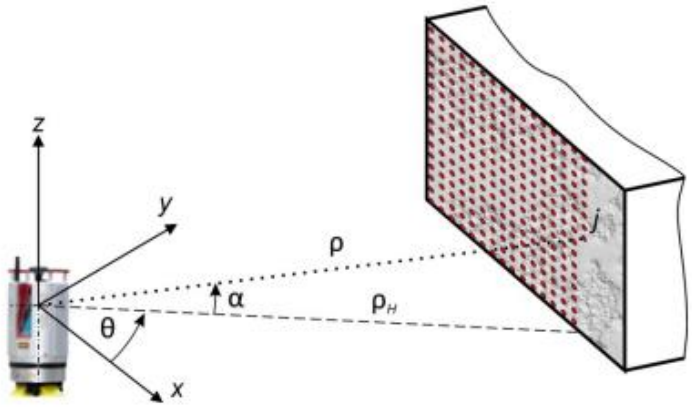
Slika 3.8. Kriterij prihvaćanja ili odbacivanja pojedinih točaka u LOAM Livox algoritmu

Na točno pozicioniranje instrumenta u svakom trenutku također se vodi računa. Provjera i optimizacija položaja vrši se na način da se iterativno uspoređuju podaci najnovije vremenske epohe, sa prijašnjima. Uspoređuju se plohe i rubovi u različitim epohama snimanja. Kako bi se izbjeglo zamućenje zbog kretanja postepeno se obrađuju dijelovi cijelog oblaka točaka i vrši se linearna interpolacija. Uz sve to u algoritam još ulazi i odbacivanje grubih grešaka i eliminacija pokretnih objekata.

4. REZULTATI LASERSKOG SKENIRANJA

Lasersko skeniranje nije zamjena za postojeće tehnike geodetskog snimanja, ali je varijanta koja se može upotrijebiti u većini geodetskih poslova. Skeniranje se odvija, kako je već navedeno, metodom registracije kuta i udaljenosti do određene točke u području snimanja. Rezultat ovakvog načina snimanja je skup trodimenzionalnih točaka, vremenski određenih, a naziva se „oblak“ točaka. Prostorna udaljenost između susjednih snimljenih točaka unutar oblaka točaka ovisi o blizini objekta snimanja i tehničkoj specifikaciji samog instrumenta. Većina današnjih skenera može snimiti vrlo guste oblake točaka, pa je tako moguće dobiti točke na snimljenom objektu međusobno udaljene tek jedan milimetar. Oblak točaka može uz svoje prostorne, relativne ili apsolutne koordinate sadržavati i intenzitet RGB (engl. *Red, Green, Blue*) boje reflektirane površine. Reflektira li se laserska zraka od zelenog lista na drvetu, ta točka će uz pripadajuće koordinate sadržavati i podatak o boji i intenzitetu reflektirane zrake. Budući da se laserskim skenerom često prikupi i više milijuna točaka po stajalištu, vođenje detaljne skice je nepotrebno, jer se iz oblaka točaka može dobiti i više nego dovoljno informacija za identifikaciju svih snimljenih objekata, te izradu plana situacije. Kao primjer izvrsno može poslužiti snimljena (skenirana) cesta, te uz cestu postavljena ploča na kojoj piše ime ulice. Iz oblaka točaka lako je pročitati naziv ulice na ploči, jer je uz nekoliko stotina točaka dobiven oblik i informacija o intenzitetu gdje će se svaka boja drugačije reflektirati (npr. bijela slova na plavoj podlozi) (Lasić 2008.).

Rezultati mjerenja kod TLS su kosa dužina δ, horizontalni pravac θ, vertikalni kut α i intenzitet povratnog signala I (Slika 4.1.). U geometrijskom smislu, skeniranjem objekta nastaje skup točaka velike gustoće i pravilnog rasporeda koji se naziva oblak točaka. Stoga svaka točka u oblaku točaka je jednoznačno definirana trima sfernim koordinatama δ, θ i α. To je izvorni rezultat opažanja odnosno tzv. „sirov podatak“. Međutim, u praksi kod komercijalnih TLS sistema rezultatima se smatraju pravokutne koordinate (x, y, z) i intenzitet registriranog zračenja I (Pejić, 2013).



Slika 4.1. Princip skeniranja i rezultati mjerenja kod terestričkog laserskog skeniranja (Pejić, 2013)

Svakoj točki također može biti pridružen i niz svojstava zvanih komponente, koje sadrže vrijednost koja opisuje točku. Te vrijednosti komponenti mogu se koristiti za klasificiranje različitih dijelova zbirke točaka sadržanih u geometriji oblaka točaka. Sve komponente imaju naziv i vrstu. Moguće vrste komponenti su Real64, Real32, UInt64, UInt32, UInt16, UInt8, Int64, Int32, Int16, Int8 i String. Numeričke komponente dodatno mogu imati faktor skaliranja i pomaka; u takvim slučajevima, korištena vrijednost komponente bit će *vrijednost * faktor skaliranja + faktor pomaka*. Iako komponente mogu imati bilo koji naziv, postoji nekoliko uobičajenih komponenti koje postoje u nekoliko formata i prikazane su u Tablici 4.1. (URL 18).

Tablica 4.1. Neke od čestih komponenti koje opisuju točke u oblaku točaka (URL 18)

Naziv komponente	Opis	Naziv komponente	Opis
x	X komponenta geometrije	**angle**	Kut zrake kojom je točka snimljena
y	Y komponenta geometrije	**flight_line**	Broj linije leta u kojoj je točka otkrivena
z	Z komponenta geometrije	**flight_line_edge**	Specificira leži li točka na rubu skena (snimke), uzduž linije leta
intensity	Intenzitet povratnog signala	**scan_direction**	Smjer u kojem gleda lasersko zrcalo u trenutku snimanja točke

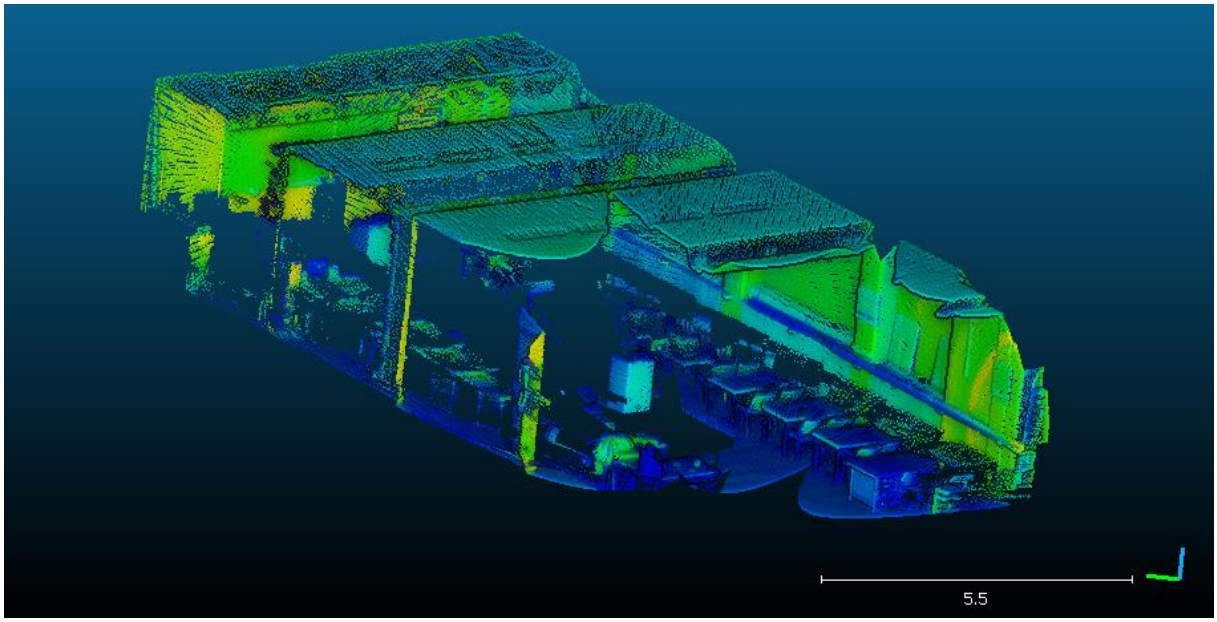
color_red	Vrijednost crvenog kanala točke	**point_source_id**	Broj identifikator datoteke
color_green	Vrijednost zelenog kanala točke	**posix_time**	Prije korišteno za vrijeme kao broj sekundi proteklih od UTC 1.1.1970.
color_blue	Vrijednost plavog kanala točke	**user_data**	Podaci se koriste prema nahođenju korisnika
classification	Klasa točke. Kategorizira točke u polja, kao npr. tlo, građevina, voda…	**normal_x**	X komponenta normale
return	Broj povratnih signala u odnosu na odaslane	**normal_y**	Y komponenta normale
gps_time	Broj sekundi od početka tjedna	**normal_z**	Z komponenta normale
gps_week	Broj tjedna počevši od 6.1.1980.		

4.1. LAS format

LAS format podataka je javni format namijenjen međusobnoj razmjeni 3D podataka oblaka točaka između korisnika. Iako je razvijen prvenstveno za razmjenu LiDAR podataka, ovaj format podržava razmjenu bilo kojih 3D x,y,z podataka. LAS je binarni format datoteka koji koristimo kao alternativu, međusobno nekompatibilnim, formatima razvijenim od strane različitih proizvođača instrumenata ili generičkom ASCII formatu koji je široko rasprostranjen. Problem kod različitih formata kod različitih proizvođača instrumenata je nemogućnost njihovih kombiniranja i međusobnih prijenosa. Kod prijenosa ASCII datoteka prisutna su dva velika problema: prvi su performanse iz razloga što učitavanje i interpretacija ASCII elevacijskih podataka mogu biti vrlo spori, a same datoteke jako velike čak i za male količine podataka; dok je drugi problem taj da se gube sve informacije specifične za LiDAR podatke. LAS format datoteke je binarni format koji održava informacije specifične za „lidarsku prirodu podataka“, a nije suviše kompleksan i velik (URL 19).

Od 2013. godine LiDAR korisnici imaju mogućnost prilagođavati LAS format specifičnim potrebama svojih aplikacija. Taj se mehanizam naziva LAS Domain Profile i on je izvedenica temeljnih LAS v1.4 specifikacija kojima se mogu dodavati nove klase i atributi točaka, ali ne i uklanjati ili izmjenjivati postojeći. U ožujku 2019. održavanje LAS specifikacija preseljena je

na GitHub (URL 19). Na Slici 4.2. prikazan je snimak učionice 11, Geodetskog fakulteta, u LAS formatu.



Slika 4.2. Prikaz snimljenog zatvorenog prostora u LAS formatu

4.2. Oblak točaka (engl. *PointCloud* - PCD)

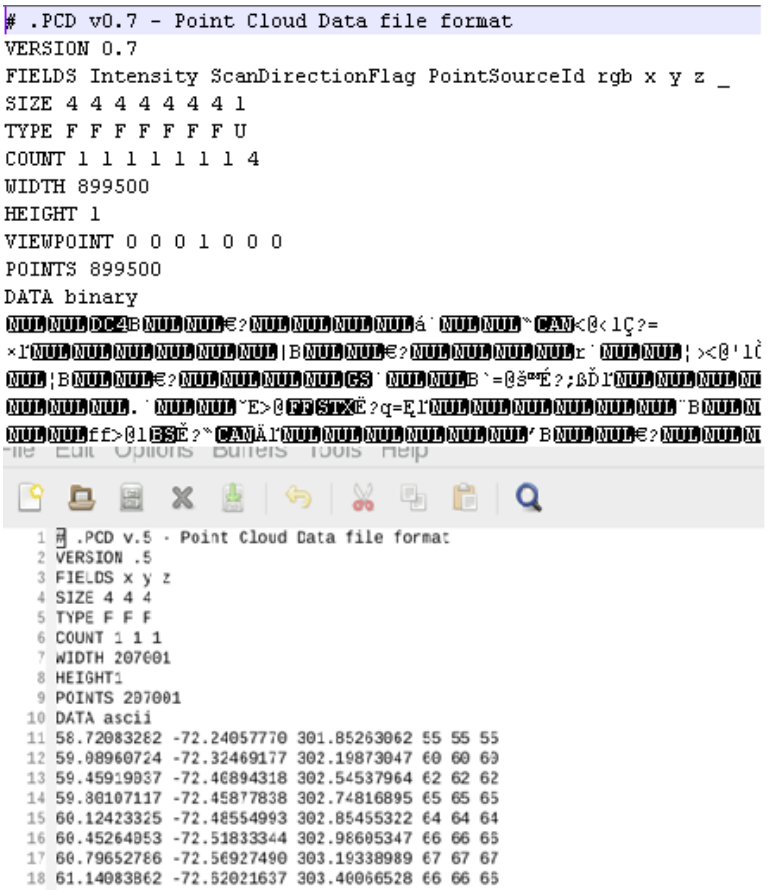
PCD (engl. *PointCloud*) je format za pohranu podataka nastao pod PCL-om (engl. *Point Cloud Library*), a kreiran je kako bi ispravio nedostatke prijašnjih formata, pritom ne gubeći vrijeme na ponavljajuće radnje. PCD datoteke sadrže dvije sekcije: čovjeku čitljivo zaglavlje koje definira određene karakteristike oblaka točaka spremljenog u datoteci kao što su broj, veličinu, dimenzionalnost i vrstu podataka oblaka točaka; te sekciju podataka koja može biti ASCII ili binarnog oblika što je čovjeku nečitljiv tip podataka. Zaglavlje mora biti kodirano ASCII kodom. Svako polje zaglavlja i ascii podataka točke naznačene u PCD datoteci odvojene su novim redom (\n). Zaglavlje sadrži polja navedena u Tablici 4.2.. Podaci navedeni u zaglavlju PCD datoteke moraju biti navedeni istim redoslijedom kao što su navedeni u tablici. PCD podržava samo brojčane komponente (URL 20).

Tablica 4.2. Polja zaglavlja PCD datoteke (URL 21)

Karakteristika	Opis
VERZIJA (VERSION)	Specificira verziju PCD datoteke
POLJA (FIELDS)	Specificira ime svake dimenzije i polje koje točka može imati Npr.:
FIELDS xyz	XYZ podaci

FIELDS xyz rgb	XYZ podaci + boje
FIELDS xyz normal_x normal_y normal_z	XYZ + normale plohe
VELIČINA (engl. SIZE)	Specificira veličinu svake dimenzije u bajtovima: - nepotpisani **char** ima 1 bajt - nepotpisani **short** ima 2 bajta - nepotpisani **int** ima 4 bajta - nepotpisani **double** ima 8 bajtova
TIP (engl. TYPE)	Specificira tip svake dimenzije sa jednim slovom: **I** – potpisani tipovi int8(char), int16(short) i int32(int) **U** – nepotpisani tipovi uint8(unsigned char), uint16(unsigned short) i uint32(unsigned int) **F** – floating point
Količina (engl. COUNT)	Specificira broj sadržanih elemenata u svakoj dimenziji
ŠIRINA (engl. WIDTH) Equal points	Broj točaka koji ukazuje na širinu oblaka točaka Ukoliko je oblak točaka neorganiziran, ovaj broj je jednak ukupnom broju točaka
VISINA (engl. HEIGHT) Equal 1	Broj točaka koji ukazuje na visinu oblaka točaka Ukoliko je oblak točaka neorganiziran, vrijednost ovog broja je 1
STAJALIŠTE (engl. VIEWPOINT)	Specificira točku gledišta oblaka točaka. Određeno kao translacija (txtytz) + rotacija (qwqxqyqz) → zadana vrijednost je: VIEWPOINT 0 0 0 1 0 0 0
TOČKE (engl. POINTS)	Specificira broj točaka u oblaku točaka
PODACI (engl. DATA)	Specificira tip podataka za pohranjivanje oblaka točaka (ascii i binarni tip)

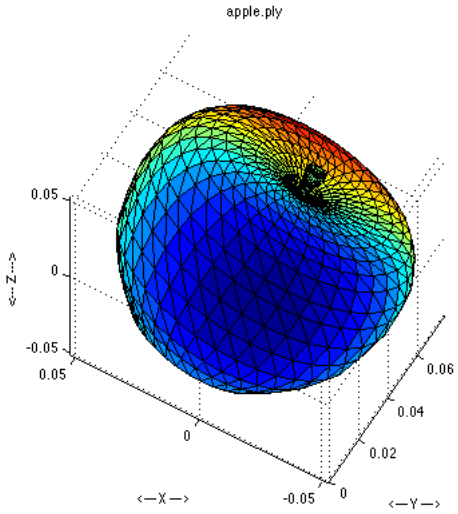
Najveća prednost PCD formata je njegova brzina i aplikabilnost, te uporaba sa PCL bibliotekom pri čemu nam ne trebaju nikakve konverzije. Mogućnost pohrane i rukovanja skupovima oblaka točaka važna je za aplikacije u stvarnom vremenu poput poboljšane stvarnosti, robotike i slično. Binarni *mmap/munmap* tip podataka je najbrži način za preuzimanje i pohranjivanje podataka na disk. Ovaj format također ima mogućnost pohranjivanja različitih tipova podataka (Dehai i dr., 2012). Primjeri tekstualne PCD datoteke sa binarnim i ASCII kodom prikazan je na Slici 4.2.



Slika 4.3. Primjer PCD datoteke – binarni kod gore; ASCII kod dolje (URL 22)

4.3. PLY format (engl. *Polygon File Format*)

PLY (engl. *Polygon File Format*; poznat i kao Stanford Triangle Format) format je format za pohranu grafičkih objekata i 3D podataka laserskih skenera koji su opisani kao kolekcija nominalno ravnih poligona (Slika 4.3.). Ideja iza nastanka ovog formata je omogućiti format koji je jednostavan i lagan za implementiranje, a ujedno i dovoljno općenit da bi se koristio na širokom rasponu modela. Nastojalo se da bude dovoljno fleksibilan kako bi predvidio i zadovoljio potrebe u budućnosti i dovoljno jednostavan da u početku ne odbije potencijalne korisnike. Postoje dvije verzije formata odnosno dva pod formata: ASCII prikaz za laki početak i binarna verzija za kompaktnu pohranu i za brzo spremanje i učitavanje. Želja je također da ovaj format potakne razmjenu grafičkih podataka među različitim programima i između grupa ljudi (URL 23).



Slika 4.4. 3D model objekta u PLY formatu (URL 24)

PLY format opisuje objekt kao kolekciju točaka (vrhova), ploha i drugih elemenata zajedno sa bojom i smjerom normale koji se mogu dodati ovim elementima. PLY datoteka sadrži opis točno jednog objekta. Ti objekti mogu biti: ručno digitalizirani objekti, poligonski objekti iz programa za modeliranje, podaci o rasponu, izopovršine iz podataka o volumenu, podaci o terenu, modeli radioznosti. Svojstva koja se mogu pohraniti s objektom uključuju: boju, normale površina, „oblike“ površine, koordinate teksture, prozirnost (transparentnost), pouzdanost podataka o rasponu i različita svojstva za prednju i stražnju stranu poligona.

Tipičan opis jednog PLY objekta je jednostavno lista (x,y,z) tripleta točaka i lista ploha koje su opisane indeksima iz popisa točaka. Svaki element u datoteci ima fiksan broj „obilježja“ koja su specifična za taj element.

Struktura formata (URL 23):

```
Header                    → Zaglavlje
Vertex List               → Popis točaka
Face List                 → Popis lica
(lists of other elements) → Popis ostalih elemenata
```

Zaglavlje uključuje opis svake vrste elementa, uključujući naziv elementa (npr. rub), koliko takvih elemenata ima u objektu i popis različitih svojstava povezanih s elementom. Zaglavlje također govori je li datoteka binarna ili ASCII. Nakon zaglavlja nalazi se jedan popis elemenata za svaki tip elemenata, redoslijedom opisanim u zaglavlju. Ispod toga je potpuni ASCII opis za objekt (Slika 4.4.).

```
ply
format ascii 1.0           { ascii/binary, format version number }
comment made by Greg Turk  { comments keyword specified, like all lines }
comment this file is a cube
element vertex 8           { define "vertex" element, 8 of them in file }
property float x           { vertex contains float "x" coordinate }
property float y           { y coordinate is also a vertex property }
property float z           { z coordinate, too }
element face 6             { there are 6 "face" elements in the file }
property list uchar int vertex_index { "vertex_indices" is a list of ints }
end_header                 { delimits the end of the header }
0 0 0                      { start of vertex list }
0 0 1
0 1 1
0 1 0
1 0 0
1 0 1
1 1 1
1 1 0
4 0 1 2 3                  { start of face list }
4 7 6 5 4
4 0 4 5 1
4 1 5 6 2
4 2 6 7 3
4 3 7 4 0
```

Slika 4.5. Struktura PLY datoteke (URL 24)

Nakon početka zaglavlja nalazi se ključna riječ „format“ i specifikacija ASCII ili binarnog formata, nakon čega slijedi broj verzije. Sljedeći je opis svakog od elemenata u datoteci poligona, a unutar svakog opisa elementa je specifikacija svojstava.

5. PRIPREMA RAČUNALA ZA LOAM LIVOX ALGORITAM

Da bi pokretanje cijelog Livox sustava bilo moguće i kako bi se dobili željeni podaci, na računalu treba izvršiti niz pripremnih predradnji. Sve potrebne instalacije i zahvati odrađeni kako bi sam LiDAR funkcionirao bit će pobliže objašnjeni u ovom poglavlju, a to su redom:

- Instalacija Linux Ubuntu 18.04.6 operacijskog sustava
- Instalacija Robotskog Operacijskog Sustava (ROS)
- Instalacija Ceres Solver-a
- Instalacija PCL biblioteke
- Instalacija drivera za Livox-Mid100 LiDAR
- Instalacija Livox-SDK paketa

5.1. Linux Ubuntu 18.04.6. LTS operacijski sustav

Kao i Windows, iOS i MacOS, Linux je operativni sustav. Svoju najširu primjenu dobio je pokrećući jednu od najpopularnijih platformi za mobilne uređaje - Android sustav. Operativni sustav je softver koji upravlja svim hardverskim resursima povezanim sa stolnim ili prijenosnim računalom. Pojednostavljeno, operativni sustav upravlja komunikacijom između softvera i hardvera. Bez operativnog sustava softver ne bi funkcionirao. Razlozi prijelaza i korištenja ovog sustava su razne njegove prednosti, a u prvom redu to je činjenica da je Linux evoluirao u jedan od najpouzdanijih računalnih ekosustava na planeti. To znači da ima puno manje prepreka od drugih sustava poput virusa, sporog rada, rušenja sustava, skupih popravaka i naknada za licence proizvoda (URL 25).

Linux ima niz različitih verzija koje odgovaraju različitim tipovima korisnika. Te se verzije još nazivaju i distribucijama. Svaka distribucija Linux-a može se besplatno preuzeti, prebaciti na prijenosni disk i sa njega instalirati na računalo. Ubuntu je jedna od distribucija Linux-a i to je jedan od računalnih operacijskih sustava sličnih Unixu. Ubuntu je nastao kao izvedenica sustava Debian GNU/Linux, koji pak mnoge temeljne komponente preuzima iz projekta GNU i koristi Linux kao jezgru (engl. *kernel*) operacijskog sustava. Ubuntu za osobna računala također standardno sadrži grafičko korisničko sučelje (engl. *Graphical User Interface - GUI*) (URL 26).

Projekt Ubuntu vodi tvrtka Canonical Ltd. koju je osnovao južnoafrički biznismen Mark Shuttleworth, no kako se radi o slobodnom softveru, u projektu sudjeluju i mnogi neovisni programeri i suradnici povezani u zajednicu Ubuntu (engl. *Ubuntu community*). Naziv Ubuntu dolazi od afričkog filozofskog koncepta „ubuntu“ koji se može prevesti kao "humanost prema drugima". Ubuntu najviše pažnje posvećuje lakoći instaliranja i korištenja, slobodi od bilo kakvih ograničenja te ustaljenom rasporedu izdavanja novih inačica (URL 27).

Za ovaj rad korištena je Ubuntu 18.04.6. LTS (engl. *Long Term Support - LTS*) verzija.

5.1.1. Često korištene naredbe

Za komunikaciju sa računalom, a time i sa LiDAR-om koristi se terminal *„bash“*. To je Ubuntu-ov ekvivalent *„command prompta“* ili *„shell-a“* na Windowsima i upisivanjem naredbi u njega upravlja se i ima uvid u cijeli projekt. U nastavku slijedi uvid i pojašnjenje pojedinih često korištenih naredbi koje su vrlo korisne za bolje razumijevanje i snalaženje u sustavu (Tablica 5.1.).

Tablica 5.1. Često korištene naredbe i kratice Ubuntu-a (URL 28)

Naredba	Značenje	Funkcija naredbe
~		Oznaka za „home“ folder
$	**S**tandard user	Oznaka koja kazuje kako je korisnik u sustav prijavljen kao standardni korisnik sa standardnim privilegijama i mogućnostima
sudo		Davanje administratorskih ovlasti
pwd	**P**rint (present) **W**orking **D**irectory	Pokazuje apsolutni put do trenutnog direktorija
cd	**C**hange **D**irectory	Mijenjanje trenutnog direktorija u onaj koji slijedi nakon naredbe; ukoliko se upisuje bez nastavka vraća korisnika u home folder
cd ..		Diže korisnika jednu razinu više u strukturi direktorija
ls	**L**ist **S**torage	Prikaz svih podataka u trenutnom direktoriju
ls -a	**L**ist **S**torage + **a**ll	Prikaz svih podataka u trenutnom direktoriju, uključujući skrivene stavke
ls -l	**L**ist **S**torage + **l**ong	Prikaz svega što se nalazi u direktoriju (osim skrivenih datoteka) uz detaljan prikaz značajki (atributa) svake datoteke
ls -la	**L**ist **S**torage + **l**ong **a**ll	Prikaz svega što se nalazi u direktoriju, uključujući skrivene datoteke, uz detaljan prikaz značajki (atributa) svake datoteke
locate [naziv]		Pokazuje put do direktorija u kojem se objekt nalazi

which [naziv]		Lociranje naredbe
↑		Listanje posljednjih naredbi
history		Ispisuje zadnjih 1000 naredbi
whatis [naziv]		Jednostavan opis neke naredbe ili funkcije
apropos [naziv]		Prikazuje sve naredbe povezane sa [naziv]
man [naziv]	**man**ual	Opširan opis naredbe ili funkcije
mkdir [opcija] naziv_direktorija	**m**a**k**e **dir**ectory	Stvaranje novog direktorija
chmod		Mijenjanje dopuštenja (čitanje, pisanje, izvršavanje)
killall [naziv]		Završi [naziv] proces
Kratice na tipkovici		
Ctrl + Alt + T	Otvaranje novog terminala	
Ctrl +Alt + D	Minimizira sve prozore tekućih procesa i pokazuje radnu površinu.	
Ctrl + C	Izlaz iz naredbe koja se izvršava (engl. *quit*)	
Ctrl + D	Signalizacija terminalu da nema više ulaznih podataka	
Ctrl + Shift + [+] / [-]	Povećanje/smanjenje teksta terminala	

5.2. Robotski operativni sustav - ROS

U današnje vrijeme roboti svih vrsta su sve prisutniji u čovjekovoj svakodnevnici, te im pomažu u obavljanju svakojakih zadataka. Međutim, roboti nemaju sposobnost samostalnog razmišljanja i zaključivanja već za rješavanje bilo kojeg zadatka trebaju dobiti detaljne upute do najsitnijeg detalja. Tako naprimjer, već i za trivijalan zadatak poput dohvaćanja nekog predmeta robotu treba pojasniti niz koraka. Prvo, robot mora razumjeti zadatak koji treba izvršiti. Sam zadatak zadaje se glasovnim putem ili nekim drugim korisničkim sučeljem poput SMS-a, e-maila ili sličnog. Zatim se robot mora orijentirati u prostoru, mora locirati predmet te izraditi neku vrstu plana kako bi došao do predmeta. Nakon što dođe do predmeta, mora znati na koji način ga može dohvatiti i što s njim treba učiniti. Za rješavanje svakog od tih pod problema robotu treba nekakav senzor kojim dobiva informacije o okolini, te nekakav motor kojim se kreće i fizički izvršava naredbe. Kako bi povezali sve te dijelove robota potreban nam

je robotski operativni sustav. Putem tog sustava čovjek programira robota i uči ga, te robotski operativni sustav praktički postaje robotu mozak.

S obzirom da je ROS vrlo složen i značajan sustav i njegovo razumijevanje je od iznimne važnosti za kasnije korištenje LiDAR-a u sklopu cjelokupnog sustava, biti će opširnije objašnjen u nastavku.

U službenoj distribuciji ROS-a postoje mnogobrojni softverski paketi koje su napisali i koje održava veliki broj razvojnih programera. ROS pruža mogućnost korištenja svih dijelova robotskog softverskog sustava kako ih ne bi morali zasebno pisati. Strukturiran je kao veliki broj malih računalnih programa koji brzo prenose poruke jedni drugima. To je struktura koja omogućuje stvaranje generičkih modula koji se primjenjuju na široke klase robotskih hardvera i softvera, te olakšava dijeljenje koda i ponovno korištenje u globalnoj robotičkoj zajednici. Takva je paradigma odabrana kako bi potaknula ponovno korištenje robotskog softvera izvan određenog robota i okruženja korištenog pri programiranju robota. Od 2007. godine kada je Morgan Quigley na Sveučilištu Stanford započeo projekt ROS-a do danas objavljeno je 13 verzija ROS-a koje se još nazivaju ROS distribucije (Lentin, 2015).

U ovom radu korištena je pretposljednja distribucija objavljena u svibnju 2018. godine, ROS Melodic Morenia, koja se preporuča za Ubuntu 18.06.4 LTS.

5.2.1. Karakteristike ROS-a

ROS se sastoji od niza dijelova poput skupa upravljačkih programa (engl. *drivers*), zbirke temeljnih robotskih algoritama, računalne infrastrukture (za upravljanje podataka, povezivanje komponenti robotskog sustava i ugradnju vlastitih algoritama), skupa alata za vizualizaciju i ROS ekosustava. Skup upravljačkih programa omogućuje čitanje podataka sa senzora i slanje naredbi motorima i drugim aktuatorima. Rastuća zbirka temeljnih robotskih algoritama omogućuje izradu karata svijeta, navigaciju po njima, predstavljanje i tumačenje senzorskih podataka, planiranje pokreta, manipulaciju objektima i slično. ROS je postao vrlo popularan u istraživačkoj zajednici robotike i u njemu je sada dostupan veliki broj vrhunskih najsuvremenijih algoritama. Računalna struktura ROS-a omogućuje premještanje podataka, povezivanje različitih komponenti složenog robotskog sustava i ugradnju vlastitih algoritama. Inherentno je distribuiran i omogućuje neprimjetno dijeljenje opterećenja na više računala. Za vizualizaciju stanja robota i algoritama ROS ima veliki skup alata koji nam također pomažu pri otklanjanju pogrešaka u softveru robota. Konačno, ROS ekosustav obuhvaća opsežan skup resursa, kao što je *wiki* koji dokumentira mnoge aspekte ROS okvira. ROS platformu karakteriziraju sljedeća svojstva (Lentin, 2015; Quigley i dr., 2015):

Međuprocesna komunikacija omogućuje da poruke prolaze kroz sučelje koje prenosi poruku za komunikaciju između dva programa ili procesa. Primjerice, kamera obrađuje sliku u kojoj pronalazi koordinate, a zatim se te koordinate šalju procesu koji je odgovoran za daljnju obradu podataka. Ova značajka važna je za programiranje robota.

Značajke poput operacijskog sustava suprotno od onoga što ime kaže, ROS nije pravi operacijski sustav. To je meta operacijski sustav koji pruža neke funkcionalnosti operacijskog sustava. Te funkcionalnosti uključuju višedretvenost (engl. *multithreading*), kontrolu uređaja na niskoj razini, upravljanje paketima i apstrakciju hardvera.

Upravljanje paketima koji imaju svoj izvorni kod, konfiguracijsku datoteku ili podatkovnu datoteku za određeni zadatak. Oni se mogu distribuirati i instalirati na druga računala.

Programska podrška i alati na visokoj razini koji se koriste u programiranju robota. ROS podržava popularne programske jezike uključujući C++, Python i Lisp. Postoji eksperimentalna podrška za jezike kao što su C#, Java, Node.js itd. Kompletan popis nalazi se na idućoj adresi: *https://wiki.ros.org/Client%20Libraries/.*

Klijentske knjižnice za navedene programske jezike u kojima svaki razvojni programer može dobiti ROS funkcionalnosti na navedenim jezicima. Na primjer, ako korisnik želi implementirati aplikaciju za Android koja koristi ROS funkcionalnost, može koristiti knjižnicu rosjava. ROS također pruža alate za izradu robotskih aplikacija. Pomoću njih možemo izgraditi mnoge pakete s jednom naredbom. Ova fleksibilnost pomaže programerima potrošiti manje vremena u stvaranju sustava za izgradnju svojih aplikacija.

Dostupnost biblioteka treće strane Okvir ROS-a je integriran s najpopularnijim knjižnicama treće strane. Na primjer, OpenCV integriran je za robotski vid, a PCL (engl. *PointClouds*) je integriran za percepciju 3D robota. Ove knjižnice čine ROS snažnijim te omogućava programeru izgradnju moćnih aplikacija.

Gotovi algoritmi (engl. *off-the-shelf algorithms*) su korisna značajka obzirom da takvi algoritmi skraćuju vrijeme programiranja prototipa robota. ROS je implementirao popularne algoritme za robotiku kao što su: PID algoritam, SLAM algoritam za simultano lociranje i kartiranje i algoritmi za planiranje puta kao što su A*, Dijkstra i AMCL.

Jednostavnost pri prototipiranju Jedna od prednosti ROS-a su navedeni gotovi algoritmi. Uz to, ROS koristi pakete koji se mogu lako upotrijebiti na bilo kojem robotu; lako je prototipirati osobi mobilni robot prilagodbom postojećeg paketa dostupnog u ROS repozitoriju. Konačno, takav način rada može smanjiti vrijeme razvoja robotskog softvera.

Ekosustav/podrška zajednici u kojoj ROS programeri diljem svijeta aktivno razvijaju i održavaju ROS pakete. Velika podrška u zajednici uključuje programere koji postavljaju pitanja vezana za ROS na platformi za upite (*https://answers.ros.org/questions/*), raspravljaju o raznim temama i objavljuju vijesti na internet forumu ROS Discourse (*https: //discourse.ros.org*) i slično.

Opsežni alati i simulatori ROS je izgrađen s mnogim naredbenim i GUI alatima za uklanjanje pogrešaka, vizualizaciju i simulaciju aplikacija u robotici. Ovi alati su vrlo korisni

za rad s robotom. Na primjer, alat Rviz koristi se za vizualizaciju pomoću fotoaparata, laserskih skenera, inercijskih mjernih jedinica i tako dalje. Za rad s robotskim simulacijama, postoje simulatori poput Gazebo simulatora.

ROS jednadžba Projekt ROS može se definirati u jednoj jednadžbi:

ROS = komunikacija + alati + mogućnosti + ekosustav

Ukratko, ROS je kombinacija komunikacije, alata, sposobnosti i ekosustava/zajednice. ROS se sastoji od triju konceptualnih razina: datotečnog sustava (engl. *Filesystem Level*), razine nadzora i upravljanja (engl. *Computation Graph Level*), te razine zajednice (engl. *Community Level*). Svaka razina biti će opisana u nastavku.

5.2.2. ROS sustav datoteka (engl. *ROS Filesystem Level*)

ROS razina sustava datoteka služi nam za oblikovanje i stvaranje minimalnog broja potrebnih resursa potrebnih za rad sustava. Mape i datoteke koje susrećemo na disku prilikom korištenja ROS-a su sljedeće (Slika 5.1) (URL 29):

Metapaketi (engl. *Metapackages*) su specijalizirani paketi u ROS-u. Oni ne instaliraju datoteke i ne sadrže nikakve testove, kodove, datoteke ili ostale stvari koje pronalazimo u paketima. Služe samo za predstavljanje skupina povezanih paketa, a najčešće se koriste kao oznaka koja osigurava kompatibilnost među paketima koji se modificiraju tijekom programiranja i implementacije.

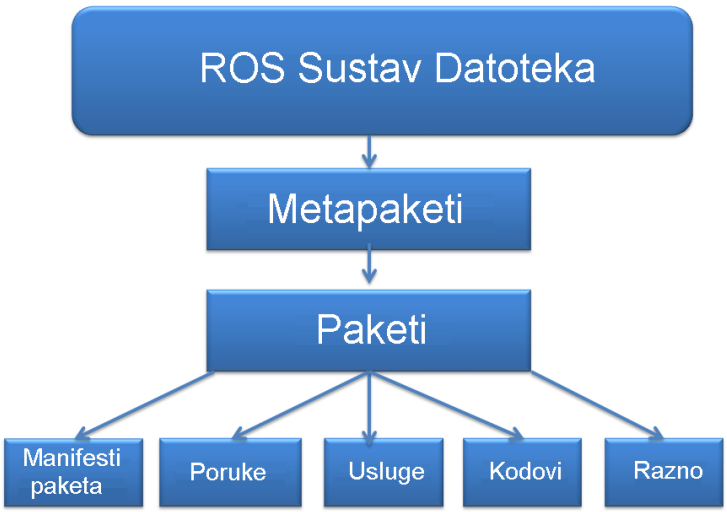
Paketi (engl. *Packages*) su glavna jedinica organizacije softvera u ROS-u i najviša stavka izrade i izdavanja. Softver u ROS-u organiziran je u pakete. Oni mogu sadržavati ROS izvršne procese, tzv. čvorove (engl. *Nodes*), ROS nezavisnu knjižicu (engl. *Library*) o kojoj ovisi, skupove podataka, konfiguracijske datoteke, dijelove ostalih softvera „treće strane“ (engl. *third-party*) i sve ono drugo što logično predstavlja koristan modul. Cilj paketa je pružiti korisniku funkcionalnost na jednostavan način i čuvanje potrebnih resursa za ponovno pokretanje. Općenito, ROS paketi slijede načelo zvano *„Goldilocks“*, što znači da pružaju dovoljnu funkcionalnost za korištenje, ali pritom vodeći računa da paketi nisu preveliki i kompliciranu za korištenje od strane drugih softvera. Mogu se stvoriti ručno ili pomoću alata poput *catkin_create-pkg* i u sebi sadrže datoteku package.xml.

Manifesti paketa (engl. *Package manifest*) su XML datoteke nazvane package.xml koji moraju biti uključeni u svaku „korijensku mapu“ (engl. *root-folder*) catkin-kompatibilnog paketa. Oni definiraju metapodatke paketa kao što su naziv, verzija, opis, informacije o licenci, autori, održavatelji i ovisnosti o drugim catkin paketima.

Repozitoriji (engl. *Repositories*) su zbirke paketa koji dijele zajednički sustav upravljanja značajkama (engl. *Version Control System – VCS*). Paketi koji dijele VCS dijele istu verziju i mogu biti objavljeni zajedno. Repozitoriji također mogu sadržavati i samo jedan paket.

Tipovi poruka (engl. *Message (msg) types*) su opisi poruka, pohranjeni u *my_package/msg/MyMessageType.msg* i definiraju strukture podataka za poruke poslane u ROS-u.

Tipovi usluga (engl. *Service (srv) types*) su opisi usluga, pohranjeni u *my_package/srv/MyServiceType.srv* i definiraju strukture podataka za komunikaciju (zahtjeva i odgovora) za servise u ROS-u.



Slika 5.1. Shematski prikaz razine sustava datoteka (engl. Filesystem level) (URL 29)

5.2.3. Razina nadzora i upravljanja (engl. *ROS Computation Graph Level*)

Sustav za nadzor i upravljanje predstavlja partnersku mrežu *(engl. peer-to-peer network)* ROS procesa (engl. *nodes*) koji obrađuje podatke - ROS graf. Razina nadzora i upravljanja zadužena je za komunikaciju između procesa i sustava. Osnovni koncept takvog sustava su čvorovi, glavni čvor ili glavni poslužitelj (engl. *Master*), poslužitelj parametara, poruke, usluge, teme i spremnici. Svi oni zajedno šalju podatke sustavu za nadzor i upravljanje. Takav koncept sustava implementiran je u *ros_comm* repozitoriju. Sljedeći koncepti su implementirani u *ros_comm* repozitorij (Lentin, 2015):

Čvorovi (engl. *nodes*) su procesi koji izvršavaju računanja (zadatke). Svaki čvor napisan je koristeći ROS klijentske knjižice *roscpp* (C++) i *rospy* (Python). Koristeći aplikacijsko programsko sučelje knjižice mogu se implementirati različite vrste komunikacijskih metoda u ROS čvorove. Upravljanje robotskim sustavima nije jednostavan proces, te se on sastoji od više različitih odvojenih zadataka. Sustavu je potrebno koristiti više čvorova odjednom kako bi mogao funkcionirati kao cjelina. Komunikacijske metode ROS-a omogućuju povezivanje i razmjenu podataka između čvorova. Neke od prednosti ovakvog načina funkcioniranja su smanjena kompleksnost koda, koji se dijeli na puno manje jednostavnijih dijelova. Također, prilikom eventualne pogreške, zahvaćen je samo jedan dio cijelog sustava odnosno jedan čvor, dok ostali i dalje mogu funkcionirati i služiti svojoj svrsi.

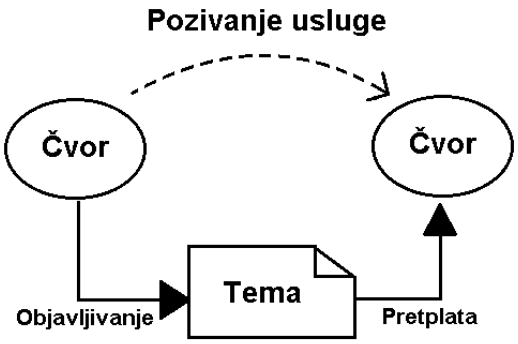
Glavni poslužitelj (engl. *Master*) omogućuje registraciju imena i traženje unutar računskog grafa. Bez glavnog poslužitelja čvorovi se ne mogu međusobno pronalaziti, izmjenjivati poruke ili dozivati usluge. On navodi izdavače i pretplatnike (engl. *publishers and subscribers*) na teme i usluge koji su im od koristi. Jednom kad određeni čvorovi stupe u kontakt oni komuniciraju samostalno na principu *„peer-to-peer“*.

Poslužitelj parametara (engl. *Parameter Server*) omogućava pohranjivanje podataka na središnjem mjestu (engl. *Central location*). Pomoću njega moguće je konfigurirati čvorove tijekom rada ili mijenjati radne parametre čvora. Poslužitelj parametara dio je glavnog poslužitelja ili *„master-a“*.

Poruke (engl. *Messages*) služe za međusobnu komunikaciju čvorova. Poruka je struktura podataka koja sadrži klasificirana polja u kojima je set podataka koji se šalju drugom čvoru. Podržava standardne tipove podataka kao što su *integer, floating point, boolean* i sl., kao i nizove jednostavnih tipova.

Teme (engl. *Topics*) su imena koja se koriste za identificiranje sadržaja poruka. Poruke se izmjenjuju na principu objavljivanje-pretplata (engl. *Publisher–Subscriber*), dok čvor šalje poruku objavljivanjem iste na određenu temu. Čvor kojeg interesira određena vrsta podataka pretplatit će se na adekvatnu temu. Jedna tema može istodobno imati više objavljivača i pretplatnika, dok jedan čvor može objavljivati i/ili pretplatiti se na više tema odjednom. Općenito, objavljivači i pretplatnici nisu svjesni međusobnog postojanja. Ideja je odvojiti proizvodnju informacija od njihove potrošnje. Na temu možemo gledati kao na tipični skup poruka. Svaki skup ima svoje jedinstveno ime i svako se sa njim može povezati za slanje ili primanje poruka ukoliko je odgovarajuće vrste.

Usluge (engl. *Services*) su posrednici pri komunikaciji zahtjev-odgovor. Model objavljivanja-pretplate je vrlo fleksibilna komunikacijska paradigma, ali njegov višestruki i/ili jednosmjerni prijenos informacija nije prikladan za interakciju na relaciji zahtjev-odgovor (engl. *Request and response*), koja je često potrebna u distribuiranom sustavu (Slika 5.2.). Servisi za zahtjev-odgovor generiraju par poruka različite strukture za zahtjev i za odgovor. Poslužiteljski čvor nudi imenovanu uslugu, a klijent koristi tu uslugu slanjem poruke zahtjeva i čeka odgovor. Kada se objavljuju teme, šalju se podaci na mnogo načina, ali kada se šalje zahtjev čvoru i želi se odgovor iz čvora, to se ne može učiniti sa temama. Servisi pružaju mogućnost interakcije s čvorovima. Također, usluge moraju imati jedinstveno ime. Kada čvor ima uslugu, svi čvorovi mogu komunicirati s njom, zahvaljujući ROS klijentskim knjižnicama.



Slika 5.2. ROS koncept rada

Spremnici (engl. *Bags*) su format u kojem se spremaju i reproduciraju podaci ROS poruka. Oni su važan mehanizam za pohranu podataka kao što su podaci sa senzora koje može biti teško prikupiti, ali su nužni za razvijanje i testiranje robotskih algoritama. Različiti alati nam omogućuju pohranu, obradu, analizu i vizualizaciju podataka. Spremnici su vrlo korisna značajka kada se radi sa kompleksnim robotskim mehanizmima.

ROS glavni poslužitelj ili *Master* djeluje kao servis imena u ROS grafu. On pohranjuje informacije o registriranim temama i uslugama u sustavu ROS čvorova. Čvorovi komuniciraju s *Master*-om kako bi prijavili svoje podatke u sustav i uspješno se registrirali u mrežu. Budući da čvorovi komuniciraju s *Master*-om oni mogu primati informacije o drugim registriranim čvorovima i ako je potrebno povezati se s njima. Master će također obavijestiti čvorove ako se neke informacije o registraciji promjene. Čvorovi se izravno povezuju s drugim čvorovima, a Master čvor samo pruža informacije o pretraživanju, slično DNS (engl. *Domain Name System*) poslužitelju. Čvorovi koji se pretplate na temu zatražit će veze s onima koji objavljuju tu temu, te će uspostaviti vezu preko dogovorenog protokola veze. Najčešći protokol koji se koristi u ROS-u naziva se TCPROS (engl. *Transport Layer for ROS Messages and Services*) i on koristi standardne TCP/IP (engl. *Transmission Control Protocol/Internet Protocol*) pristupne točke. Ovakva arhitektura omogućava modularnost; odvojeno izvršavanje operacija, gdje su imena primarna sredstva pomoću kojih se mogu graditi veći i složeniji sustavi. Imena imaju vrlo važnu ulogu u ROS-u: čvorovi, teme, usluge i parametri imaju jedinstvena imena. Svaka ROS klijentska knjižica podržava remapiranje imena putem naredbenog retka, što znači da se program tijekom izvođenja može prilagoditi za rad na drugom dijelu ROS grafa (Slika 5.3.).

Slika 5.3. Shematski prikaz ROS razine nadzora i upravljanja (prema URL 30)

5.2.4. ROS zajednica – razina (engl. *ROS Community Level*)

ROS zajednica (engl. *ROS community*) je koncept koji omogućuje resurse koji odvojenim zajednicama omogućuju razmjenu softvera i znanja. U resurse ubrajamo (Lentin, 2015):

Distribucije (engl. *Distributions*) su kolekcije skupina paketa koji su pristupačni za instalaciju pod raznim verzijama. Imaju sličnu ulogu kao distribucije Linux-a i svih drugih softvera a to je pojednostavljenje instalacije i prikupljanja zbirki softvera. ROS distribucije održavaju dosljedne verzije u nizu softvera.

Repozitoriji (engl. *Repositories*) ROS se oslanja na udruženu mrežu repozitorija za kodove, gdje različite institucije mogu razviti i izdati vlastite softverske komponente robota.

ROS Wiki zajednica je glavni forum za dokumentiranje informacija o ROS-u. Svatko se može registrirati i doprinijeti svojom dokumentacijom, predložiti ispravke i nadopune, pisati vodiče i na mnoge druge načine.

Sustav za otklanjanje pogrešaka (engl. *Bug Ticket System*) služi za prijavljivanje pogrešaka i/ili problema pronađenih u postojećem softveru, te za predlaganje novih značajki.

Mailing liste ROS korisnika služe kao komunikacijski kanal o novim ažuriranjima za ROS i kao forum za postavljanje pitanja o softveru.

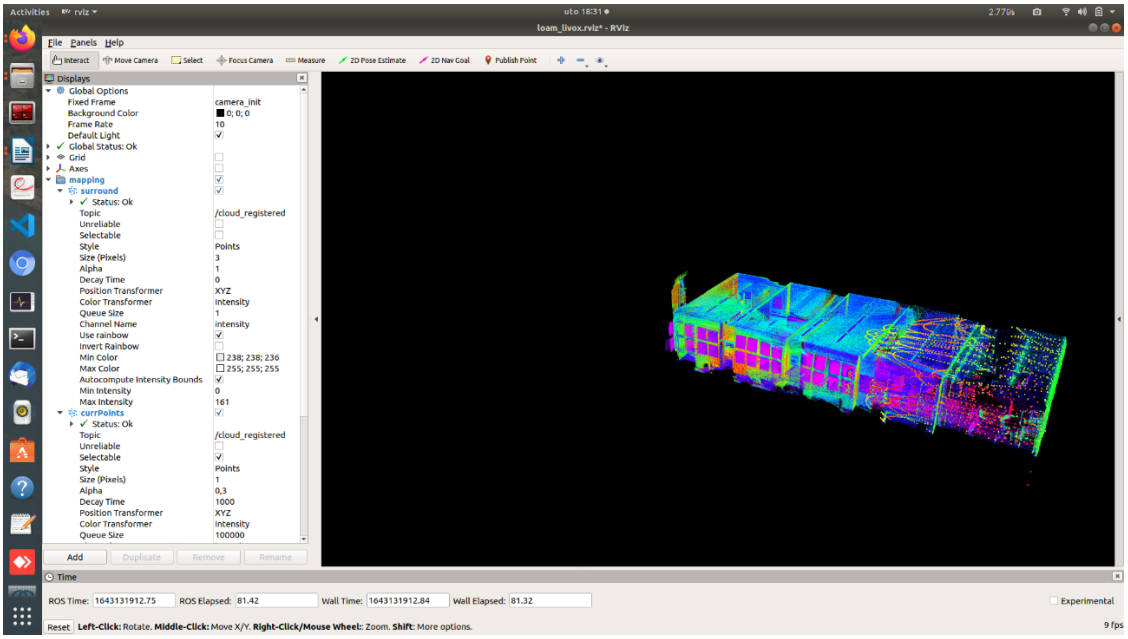
ROS odgovori (engl. *Answers*) je web-stranica koja se koristi kao resurs preko kojega korisnici postavljaju pitanja. Ako netko objavi nekakve nejasnoće u vezi ROS-a na ovoj stranici, ostatak zajednice to može vidjeti i pomoći odgovorom.

Blog Na web adresi *http://www.ros.org/news* nalaze se sva redovita ažuriranja, novosti, fotografije i videa vezana za ROS zajednicu.

5.2.5. ROS alati

Bag (spremnik) je ROS-ov format datoteka za pohranu ROS poruka. Naziv je dobio po .bag ekstenziji i igra važnu ulogu u ROS-u. Mnogi alati su nastali upravo kako bi omogućili pohranjivanje, procesiranje, analiziranje i vizualizaciju tzv. *bag*-ova. Alati još mogu služiti za, samokontrolu, izvršavanje raznih zadaća, simulaciju i uklanjanje pogrešaka. U nastavku će biti prikazani alati koji su korišteni za ovaj konkretan slučaj (URL 31).

Rviz – omogućuje 3D vizualizaciju snimljene okoline, tj prikazuje ono što robot vidi, misli i radi. Razvijanje robota bilo bi izuzetno zahtjevno bez mogućnosti da vidimo točno ono što robot zna i misli o događajima oko njega (Slika 5.4.). Rviz omogućuje gledanje okoline kroz robotove „oči“. Pomoću ovog alata mogu se kombinirati podaci raznih senzora, lasera, kamera i drugih 3D podataka u zajednički prikaz. Postoje dva glavna načina na koje svoje podatke stavljamo u Rviz-ov svijet. Prvo – Rviz razumije senzore i informacije o stanjima kao što su laserske snimke, oblaci točaka, kamere i koordinatni okviri. Također se kroz softver u snimljeni 3D prikaz također mogu slati vlastiti markeri poput kocaka, strelica i linija obojanih proizvoljnim bojama što predstavlja drugi način učitavanja podataka u Rviz. Markeri se također koriste kod planiranja kretanja za prikazivanje planiranog puta u odnosu na stvarnu snimljenu stazu, cilj, otkrivanje objekata, i kalibraciju. Kombinacija senzorskih podataka i prilagođenih umetnutih markera čini Rviz moćnim alatom pri razvijanju robotskih mogućnosti i istraživanja. Rviz također uvelike pomaže u otklanjanju pogrešaka iz mjerenja, jer je na taj način, vizualno u 3D okolini, puno lake uočiti pogreške nego što bi to bilo „skrolanjem“ kroz mnoštvo brojeva (URL 32).



Slika 5.4. RViz – ROS-ov alat za vizualizaciju

Rosbag – set alata, odnosno naredbi koji ima puno različitih namjena: npr. snimanje, prikazivanje, filtriranje, informiranje o bag datoteci. Podaci i poruke pohranjuju se u takozvanim bagovima. Rosbag ima C++ i Python aplikacijsko programsko sučelje za čitanje poruka iz i pisanje poruka u bag datoteke.

Rosbag naredbeni redak (engl. *Command-line tool*) je alat koji pruža funkcionalnost za ROS bagove. Može snimiti bag, ponovno objaviti poruku iz jednoga ili više bagova, sažeti njihov sadržaj, provjeriti definicije poruka bagova, filtrirati poruke bagova na osnovu Python izraza, komprimirati i dekomprimirati bag i obnoviti bagov „indeks“. Neke od rosbag podnaredbi prikazane su u Tablici 5.2 (URL 33). Broj podnaredbi varira, te se konstantno dodaju neke nove, ili starije izbacuju iz uporabe. Kako bi dobili uvid u sve trenutne podnaredbe rosbag-a u naredbeni redak se upisuje `rosbag -h`.

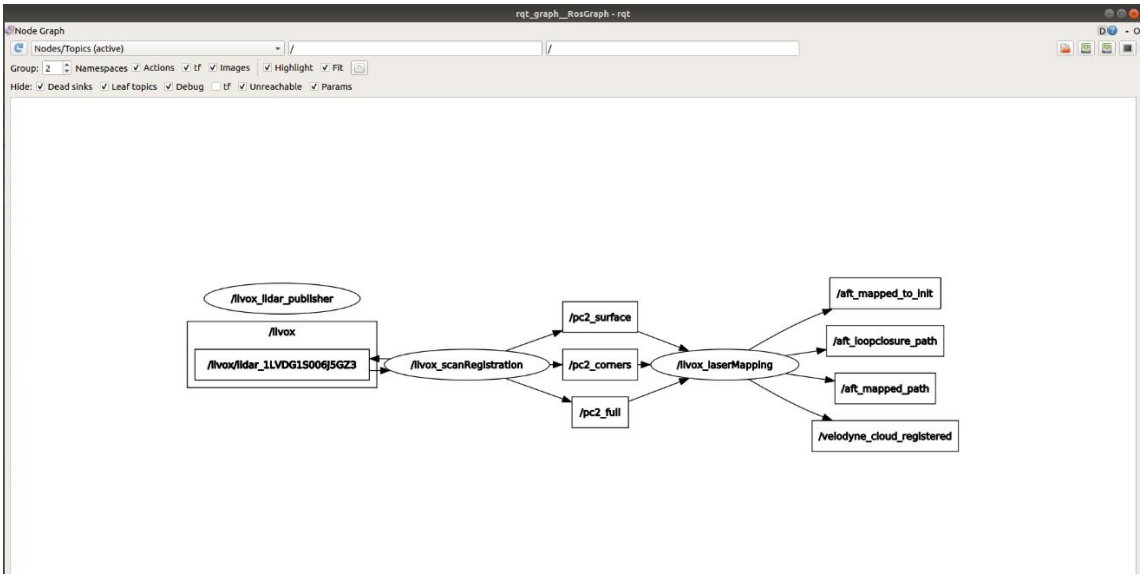
Tablica 5.2. Podnaredbe za rosbag alat (URL 33)

Naredba	Namjena
record	Snima (pohranjuje) bag datoteku sa sadržajem odabranih tema
info	Sažima sadržaj bag datoteke
play	Reproducira sadržaj jednog ili više bagova
check	Određuje može li se bag datoteka reproducirati u trenutnom sustavu ili se može preseliti
fix	Popravlja poruke u bagu kako bi ga bilo moguće reproducirati u trenutnom sustavu
filter	Konvertira bag datoteku koristeći Python ekspresije
compress	Komprimira jedan ili više bagova
decompress	Dekomprimira jedan ili više bagova
reindex	Obnavlja „indeks“ jednog ili više bagova

Rqt_bag pruža GUI dodatak za prikaz i reproduciranje datoteka ROS vrećice. To je aplikacija za snimanje i upravljanje bagovima, a neke od njezinih karakteristika su: prikaz sadržaja bag poruka, prikaz slikovnih poruka, iscrtavanje vremenskih serija vrijednosti poruka, objavljivanje/snimanje poruka na odabrane teme, te izvoz poruka u vremenskom rasponu u novu bag datoteku. Koristan je alat za pregledavanje strukture datoteke, prelazeći preko cijele datoteke, njenog dijela ili analizirajući jednu po jednu poruku. Koristeći *rqt_bag* može se na prvi pogled vidjeti koliko je tema snimljeno i koliko su često primljene poruke o svakoj temi. Također se dio postojeće datoteke može spremiti kao nova zasebna datoteka što nam omogućuje

izdvajanje samo onog dijela datoteke koji nam je koristan, a izostavljanje viška nepotrebnih podataka (Quigley i dr., 2015).

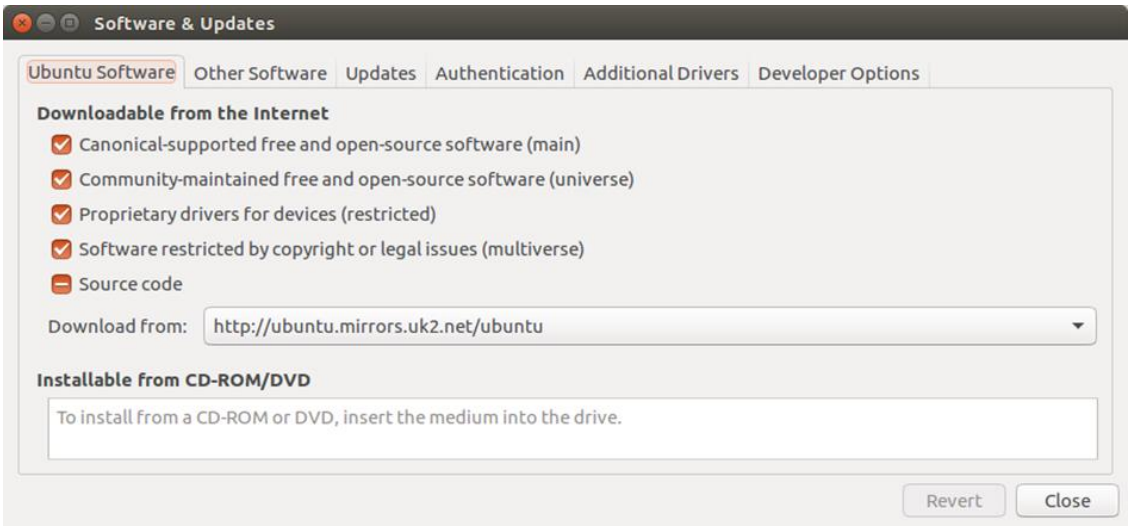
Rqt_graph je alat koji vizualizira graf trenutno aktivnih procesa u ROS-u i njihovih međusobnih odnosa i konekcija. Ovo je vrlo koristan alat i prvi korak za kontrolu rada cijelog sustava i pronalaženje eventualnih pogrešaka u radu. U *rqt_graph*-u teme su prikazane u pravokutnicima, a čvorovi u elipsama, dok poruke i parametri nisu uključeni u graf (Slika 5.5.).



Slika 5.5. Rqt_graf prikazuje trenutno aktivne procese pri pokretanju sustava

5.2.6. Instalacija ROS-a

Prvo što treba učiniti prije instalacije ROS Melodic-a je konfiguriranje Ubuntu repozitorija. To se radi na način prikazan na Slici 5.6. tako da se označe polja za omogučavanje restricted, universe, multiverse. Instalaciju ćemo izvršiti na način da slijedimo naredbe sa službene web-adrese: *http://wiki.ros.org/melodic/Installation/Ubuntu*.



Slika 5.6. Dijaloški okvir za konfiguraciju Ubuntu repozitorija (URL 34)

Nakon toga treba namjestiti računalo tako da može prihvatiti softver sa *packages.ros.org* i to na način da se u terminal upiše sljedeće:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
$(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-
latest.list'


sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb
_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Postavke ključeva:

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' -
-recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Sama instalacija započinje provjerom je li Debian «package indeks» ažuriran „up to date“:

```
sudo apt-get update && sudo apt-get upgrade
```

Postoji više različitih biblioteka i alata u ROS-u, tako da se kod instalacije nude 4 unaprijed zadane konfiguracije za instalaciju. Svaka konfiguracija sastoji se od različitih kombinacija biblioteka i alata, a također je moguće i instalirati samo individualne pakete ROS-a. Preporučuje se konfiguracija potpune instalacije koja je ovom prilikom i korištena, a koja se sastoji od instalacije ROS-a, rqt, rviz, robot-generic libraries, 2D/3D simulatori i 2D/3D percepcije.

```
sudo apt install ros-melodic-desktop-full
```

Pojedini paketi mogu se pronaći sljedećom naredbom:

```
apt search ros-melodic
```

Slijedeći korak je podešavanje okoline sustava. Kako bi se varijable ROS okoline automatski učitale pokretanjem svake nove „bash“ sesije potrebno je izvršiti nardbu:

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Do sada je instalirano sve potrebno za pokretanje osnovnih ROS paketa. Kako bi se moglo kreirati i upravljati vlastitim ROS „workspaceovima“, postoje razni alati i zahtjevi koji su distribuirani odvojeno. Na primjer, „rosinstall“ je često korišten alat naredbenog retka koji omogućava lako preuzimanje raznih izvornih stabla ROS paketa i to pomoću jedne naredbe. Za instaliranje ovog alata i drugih zahtjeva za građenje ROS paketa, potrebno je izvršiti sljedeće:

```
sudo apt install python-rosdep python-rosinstall python-rosins
tall-generator python-wstool build-essential
```

Prije no što je moguće koristiti mnoge ROS alate, potrebno je inicijalizirati „rosdep“. Rosdep omogućava jednostavno instaliranje zahtjeva sustava za izvor koji se želi kompajlirati i potreban je za pokretanje nekih osnovnih komponenta ROS-a. Ako rosdep već nije instaliran, to treba učiniti na sljedeći način:

```
sudo apt install python-rosdep
```

Dok se rosdep inicijalizira ovako:

```
sudo rosdep init
rosdep update
```

Za testiranje i bolje razumijevanje cijelog sustava na wiki stranicama nalaze se mnogi tutorijali odnosno male lekcije i primjeri zadataka koji korak po korak pojašnjavaju svaki dio ROS sustava (URL 35).

5.3. Ceres Solver

Ceres Solver je programska knjižnica C++ otvorenog koda za modeliranje i rješavanje velikih i složenih problema s optimizacijom. Može se koristiti za rješavanje dvije vrste problema:

- Problemi nelinearnih najmanjih kvadrata sa ograničenjima (engl. *Non-linear Least Squares problems with bounds constraints*).
- Opći neograničeni problemi optimizacije (engl. *General unconstrained optimization problems*).

To je dobro razvijena programska knjižica bogata značajkama i zadacima koja se koristi u razvijanju Google-a od 2010. godine (URL 36).

5.3.1. Instalacija Ceres Solver-a

Prvi korak kod instalacije Ceres Solver-a je preuzimanje datoteke *„ceres-solver.org/ceres-solver-2.0.0.tar.gz“* sa službene web stranice *http://ceres-solver.org/installation.html*. Nakon toga se trebaju instalirati sve potrebne ovisnosti. To ćemo učiniti upisivanjem sljedećih naredbi u „terminal“:

 # Cmake

```
sudo apt-get install cmake
```

 # google-glog + gflags

```
sudo apt-get install libgoogle-glog-dev libflags-dev
```

 # BLAS & LAPACK

```
sudo apt-get install libatlas-base-dev
```

 # Eigen3

```
sudo apt-get install libeigen3-dev
```

 # SuiteSparse and CXSparse (opcionalno)

```
sudo apt-get install libsuitesparse-dev
```

Nakon ovih pred koraka sve je spremno za graditi, testirati i instalirati Ceres Solver. Za to je potrebno upisati sljedeće u „terminal“:

```
tar zxf ceres-solver-2.0.0.tar.gz
mkdir ceres-bin
cd ceres-bin
```

Ovaj dio naredbe dekomprimira preuzetu mapu u datoteku *„ceres-bin“* te nas pozicionira u tu mapu. Nakon toga upisuje se sljedeće:

```
cmake ../ceres-solver-2.0.0
make -j3
make test
make install
```

Ovim naredbama instaliran je Ceres Solver i pokrenut test o uspješnosti same instalacije. Rezultat koji bi se trebao dobiti prikazan je na prije spomenutoj web stranici, te usporedba s istim može poslužiti kao samokontrola postupka.

5.4. PCL – Point Cloud Library

Biblioteka oblaka točaka (engl. *PCL – Point Cloud Library*) je velik, otvoren projekt za 2D i 3D obradu slika i oblaka točaka. PCL okvir sadrži mnogobrojne najaktualnije (engl. *„state-of-the-art“*) algoritme, uključujući filtriranje, procjenu svojstava, rekonstrukciju površine, registraciju, uklapanje modela i segmentaciju. Ovi algoritmi, primjerice, mogu biti korišteni za filtriranje odstupanja od podataka šuma, spajanje više 3D oblaka točaka, segmentiranje relevantnih „dijelova scene“, izdvajanje ključnih točaka i korištenje značajki deskriptora za prepoznavanje stvarnih objekata temeljem njihovog geometrijskog izgleda, kreiranje površina iz oblaka točaka, te u još mnoge srodne svrhe (URL 37).

PCL je pod licencom *3-clause BSD license* (engl. *Berkeley Software Distribution*) što je jedna od obitelji dopuštenih licenci za slobodni softver, koja nameće minimalna ograničenja za upotrebu i distribuciju pokrivenog softvera. To znači da je besplatan za komercijalne svrhe i u svrhe istraživanja. PCL je *cross-platform* i uspješno je sastavljen i razvijen na Linux-u, MacOS-u, Windows-ima i Android-u. Da pojednostavimo, PCL je podijeljen u niz manjih biblioteka koda, koje mogu biti „kompajlirane“ odvojeno.

5.4.1. PCL instalacija

Instalacija ove biblioteke vrlo je jednostavna a preporučene upute za istu nalaze se na *https://pointclouds.org/downloads/*. Sve što je potrebno je upisati sljedeću naredbu u terminal:

```
$ sudo apt install libpcl-dev
```

5.5. Livox SDK

Livox SDK (engl. Software Development Kit) je komplet za razvoj softvera dizajniran za sve Livox proizvode. Razvijen je na temelju C/C++ jezika slijedeći Livox SDK komunikacijski protokol i pruža jednostavno programsko aplikacijsko sučelje (engl. *Application Programme Interface*) za stil C. Pomoću Livox SDK-a korisnici se mogu brzo povezati s Livoxovim proizvodima i primati podatke u obliku oblaku točaka (URL 38).

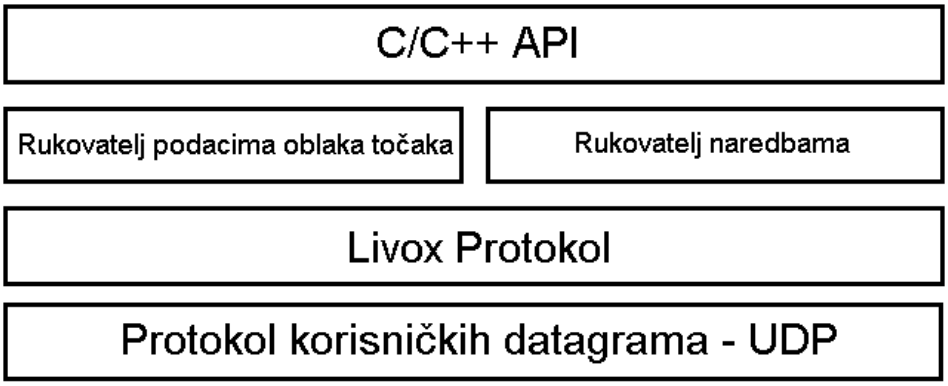
Livox SDK sastoji se od Livox SDK komunikacijskog protokola, Livox SDK jezgre, Livox SDK API-ja, Linux uzorka i ROS-a (ROS demo).

Preduvjeti za instalaciju ovog kompleta su:

- Ubuntu 14.04/Ubuntu 16.04/Ubuntu 18.04, both x86 and ARM (Nvidia TX2)
- Windows 7/10, Visual Studio 2015 Update3/2017/2019
- C++11 compiler

5.5.1. Livox SDK jezgra

Livox SDK pruža implementaciju kontrolnih naredbi i prijenos podataka oblaka točaka, kao i C/C++ API. Osnovna struktura Livox SDK jezgre je prikazana na Slici 5.7.



Slika 5.7. Pojednostavljena struktura Livox SDK jezgre

Protokol korisničkih datagrama (engl. *User Datagram Protocol - UDP*) koristi se za komunikaciju između Livox SDK i LiDAR senzora. Rukovatelj podataka oblaka točaka podržava prijenos podataka oblaka točaka, dok rukovalac naredbama prima i šalje kontrolne naredbe. C/C++ API se temelji na rukovatelju naredbama i rukovatelju podacima u oblaku točaka.

Livox LiDAR senzori mogu se spojiti na računalo izravno ili putem Livox Huba. Livox SDK podržava obje metode povezivanja. Kada su LiDAR jedinice izravno spojene na računalo, domaćin će uspostaviti komunikaciju sa svakom LiDAR jedinicom pojedinačno. A ako se LiDAR jedinice povežu s hostom preko Huba, tada host komunicira samo s Livox Hubom, dok Hub komunicira sa svakom LiDAR jedinicom.

5.5.2. Livox SDK API i njegova instalacija

Livox SDK aplikacijsko programsko sučelje (eng. *API – Application Program Interface*) pruža skup funkcija C stila koje se mogu jednostavno integrirati u C/C++ programe.

Prije nego se krene sa samom instalacijom Livox SDK paketa, na računalu je potrebno imati instaliran paket CMake 3.0.0+. Ovaj paket prethodno je instaliran u sklopu instalacije Ceres Solver-a. Livox SDK instalirati će se upisivanjem sljedećih redaka u terminal:

```
git clone https://github.com/Livox-SDK/Livox-SDK.git
cd Livox-SDK
cd build && cmake ..
make
sudo make install
```

5.6. LOAM Livox – Instalacija

Kao što je već spomenuto, LOAM Livox je algoritam koji omogućuje maksimalno korištenje potencijala Livox Mid-100 uređaja. Bez ovog algoritma, korištenjem samo softvera dobivenih uz sam uređaj, dobiva se samo trenutna vizualizacija okoline, dok je pohrana i ponovno učitavanje oblaka točaka vrlo nezgrapno riješena. Uz to, ovaj algoritam omogućuje pravilno poklapanje različitih vremenskih epoha mjerenja ukoliko se za vrijeme snimanja krećemo, te popravlja podatke nakon što se vratimo na početnu poziciju algoritmom zatvaranja petlje. Algoritam također vrši selekciju „dobrih" točaka i dodaje im karakteristična svojstva.

Algoritam je u cijelosti osmišljen za model Livox Mid-40 tako da su za korištenje Livox Mid-100 uređaja potrebne minimalne preinake koje će biti prikazane u nastavku. Promjene se rade u takozvanim „launch" fajlovima potrebnim za pokretanje cijelog sustava putem terminala, a koji se nalaze na sljedećem mjestu: `ws_livox/src/livox_ros_driver/launch` i `catkin_ws/src/loam_livox/launch` (Slika 5.8.; Slika 5.9.). S obzirom da se Livox Mid-100 sastoji od triju jedinica Livox Mid-40, u „launch" datoteci se u redu označenom na slici pod „multi_topic" izvorna „0" zamjenjuje s „1". Na taj način sustav objavljuje 3 teme *(eng. topics)* odjednom, umjesto jedne što se može provjeriti i na *rqt_graph*-u.

```
<launch>

    <arg name="lvx_file_path" default="livox_test.lvx"/>
    <arg name="bd_list" default="100000000000000"/>
    <arg name="xfer_format" default="0"/>
    <arg name="multi_topic" default="1"/>
    <arg name="data_src" default="0"/>
    <arg name="publish_freq" default="10.0"/>
    <arg name="output_type" default="0"/>
    <arg name="rviz_enable" default="false"/>
    <arg name="rosbag_enable" default="false"/>
    <arg name="cmdline_arg" default="$(arg bd_list)"/>
    <arg name="msg_frame_id" default="livox_frame"/>
    <arg name="lidar_bag" default="true"/>
    <arg name="imu_bag" default="true"/>
```

Slika 5.8. Izmjena „livox_lidar.launch“ datoteke kako bi se mogli prikupljati podaci sa više Livox jedinica odjednom

```
<launch>
    <rosparam command="load" file="$(find loam_livox)/config/performance_precision.yaml" />
    <param name="common/pcd_save_dir" type="string" value="$(env HOME)/Loam_livox" />
    <param name="common/log_save_dir" type="string" value="$(env HOME)/Loam_livox" />
    <param name="common/loop_save_dir" type="string" value="$(env HOME)/Loam_livox" />
    <param name="common/piecewise_number" type="int" value="2" />

    <param name="common/if_verbose_screen_printf" type="int" value="1"/>
    <param name="mapping/maximum_pointcloud_delay_time" type="double" value="1.0"/>

    <node pkg="loam_livox" type="livox_scanRegistration" name="livox_scanRegistration" output="screen">
     <!-- <remap from="/laser_points" to="/livox/lidar" /> -->
     <remap from="/laser_points_0" to="/livox/lidar_1LVDFCE00504521" />
     <remap from="/laser_points_1" to="/livox/lidar_1LVDFCE00504522" />
     <remap from="/laser_points_2" to="/livox/lidar_1LVDFCE00504523" />
    </node>

```

Slika 5.9. Izmjena „rosbag_mid100.launch“ datoteke

5.7. Livox ROS Driver

Livox ROS Driver je ROS paket koji nam služi za spajanje i komuniciranje Livox LiDAR uređaja i računala. Taj paket zahtjeva unaprijed instaliran Livox-SDK paket, te instaliranu ROS okolinu na računalu. Preuzimanje i gradnja *livox_ros_driver*-a vrši se preko GitHub platforme na način da upisivanjem sljedećih naredbi u terminal „kloniramo“ kod sa stranice (URL 40):

```
git clone https://github.com/Livox-SDK/livox_ros_driver.git
ws_livox/src
```

Nakon toga slijedi gradnja samog upravljačkog programa, Livox ROS Driver-a:

```
cd ws_livox
catkin_make
```

I na kraju bi bilo dobro ažurirati trenutnu okolinu ROS paketa:

```
source ./devel/setup.sh
```

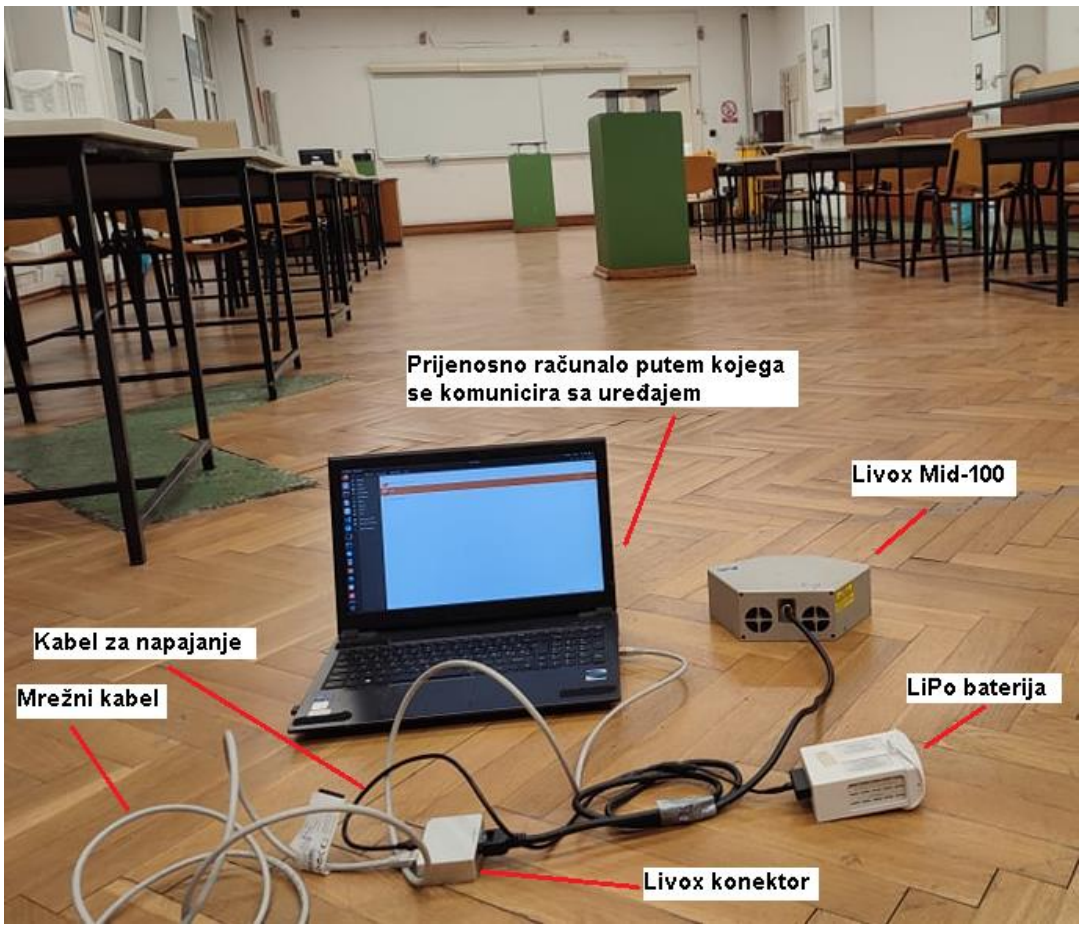
Samo pokretanje livox_ros_driver-a izvršava se naredbom:

```
roslaunch livox_ros_driver [launch file] [param]
```

u kojoj se pod `[launch file]` upisuje jedna ili više .launch datoteka koje se nalaze u `"ws_livox/src/livox_ros_driver/launch"` direktoriju, dok se pod `[param]` upisuje jedinstveni kod za emitiranje (engl. *Broadcast Code*) koji se nalazi na samom uređaju.

6. POKRETANJE LOAM LIVOX ALGORITMA

Nakon objašnjenja svake zasebne komponente ovog sustava biti će prikazano i pokretanje odnoso korištenje sustava. Kratko probno snimanje odrađeno je u zatvorenom prostoru na Geodetskom fakultetu u učionici broj 11. U prvim pokušajima pokretanja za napajanje uređaja je korištena velika automobilska baterija što je sustav činilo vrlo nezgrapnim za korištenje, no do završetka ovog rada nabavljena je lakša i manja baterija inače namijenjena dronovima. Tako je kompletni sustav postao kompaktniji i primjereniji za nošenje na većim prostorima. Uz Livox Mid-100 LiDAR uređaj, sustav se još sastoji od prijenosnog računala, Livox konvertera (engl. *Converter*), mrežnog kabla, kabla za napajanje i baterije (Slika 6.1.). Nakon pravilnog spajanja svih kablova, LiDAR se pali automatski u trenutku kad se pritisne gumb na bateriji, te je u pripravnosti do zadavanja naredbi putem prijenosnog računala. Kao sve naredbe do sada, i ove naredbe se upisuju u Linux-ov terminal *bash*, te je za početak snimanja i pohrane oblaka točaka potrebno otvoriti tri zasebna *bash* prozora.



Slika 6.1. Livox LOAM sustav spreman za korištenje

Naredbom u prvom prozoru uključuje se upravljački program odnosno *livox_ros_driver* i to na način da se upišu sljedeće naredbe:

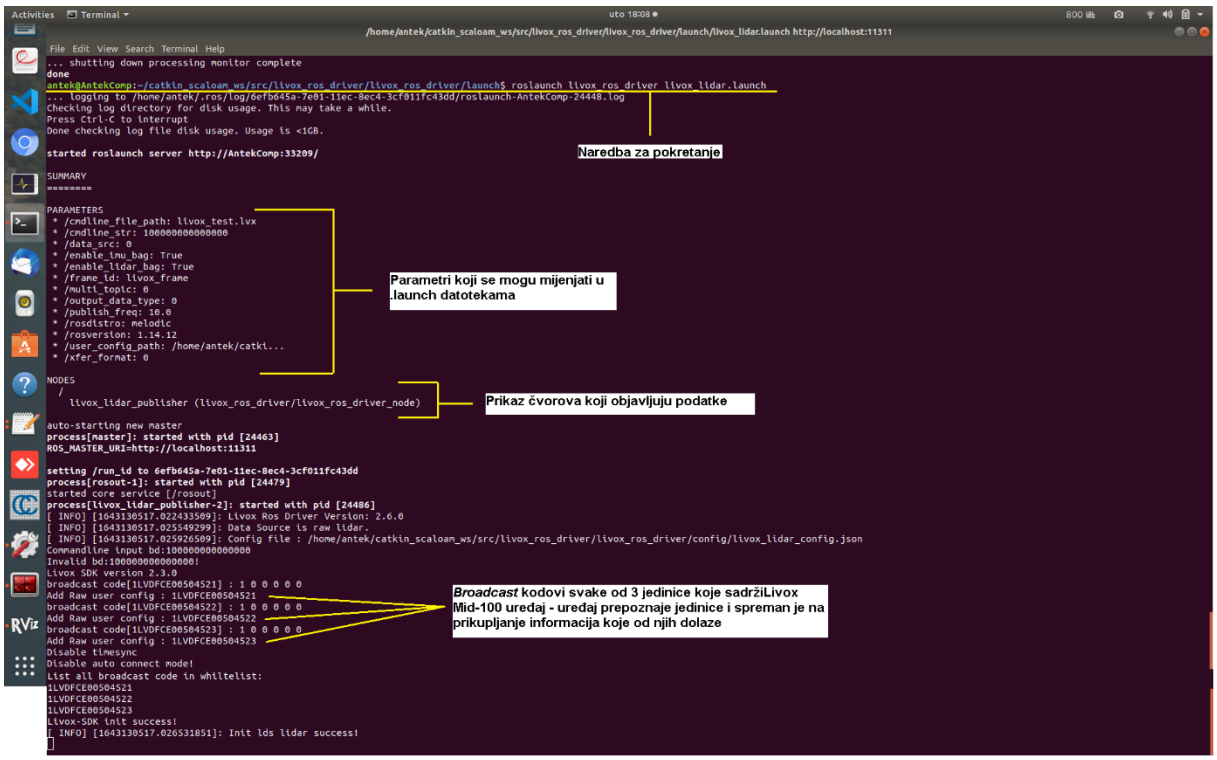
```
cd programi/ws_livox

catkin_make

source ./devel/setup.sh

roslaunch livox_ros_driver livox_lidar.launch
```

Kao što i samo ime kaže, *livox_ros driver* je upravljački program za spajanje LiDAR-a sa računalom. On služi kao „premosnica“ između hardvera i softvera i omogućuje računalu prepoznavanje i razumijevanje podataka i stanja koja pristižu sa uređaja. Bez ovog programa ne bi mogli upravljati uređajem te iskoristiti podatke koje generira (Slika 6.2.).



Slika 6.2. Prikaz bash prozora nakon pokretanja livox_ros_drivera

Upisivanjem prve naredbe računalo prepoznaje LiDAR i spremno je na početak snimanja. Međutim, na taj način mogu se prikupiti samo sirovi podaci bez ikakvih poboljšanja koje sadrži LOAM Livox algoritam. Stoga se otvara novi prozor u kojem se upisuju naredbe za pokretanje LOAM Livox algoritma što će biti objašnjeno u nastavku. Iako se na neki način i nakon upotrebe LOAM Livox algoritma dobivaju, mogli bi reći, „sirovi“ podaci koji se dalje mogu obrađivati u raznim programima za obradu oblaka točaka, taj algoritam nekim svojim funkcijama već u toku snimanja znatno poboljšava cijeli oblak, ispravlja neke greške i eliminira nepotrebne točke. To se radi na već spomenute načine poput algoritma za zatvaranje petlje,

postepenim uklapanjem novih okvira opažanja u postojeći model, te izdvajanjem korisnih točaka oblaka i pridavanjem karakteristika istim (Slika 6.3.).

```
cd programi/catkin_ws

catkin_make

source ./devel/setup.bash

roslaunch loam_livox rosbag_mid100.launch
```

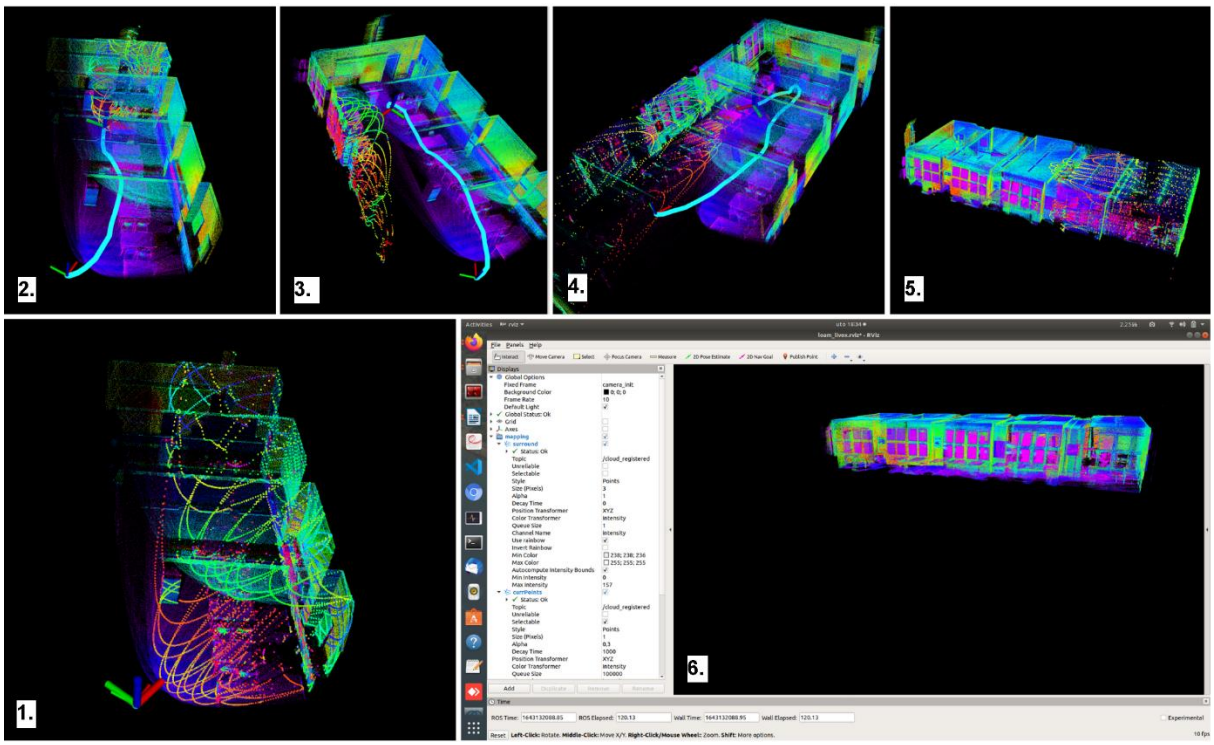
Slika 6.3. Prikaz bash prozora nakon pokretanja LOAM Livox-a

Posljednji, ne i manje bitan korak u korištenju sustava je naredba za snimanje odnosno pohranjivanje podataka. To je ROS-ova naredba koja snima cijeli tok snimanja, bilježi aktivnost svake komponente sustava, te to sprema u ROS-ovom formatu *bag*. Najčešće su najbitniji podaci koji se izvlače iz *bagova* oblaci točaka, no u bagovimase nalaze i mnogi drugi podaci o

vremenu, aktivnosti pojedinih procesa u sustavu i sl.. Naredba kojom naređujemo sustavu da bilježi svaki korak za vrijeme svoje aktivnosti je sljedeća:

```
rosbag record -a
```

Upisivanjem spomenutih naredbi pokrenut je SLAM sustav realiziran u ovom radu. Snimanje se prekida kombinacijom tipki *Ctrl + C* koja u *bash*-u znači izlaz (engl. *Quit*). Nakon kraja snimanja podaci se nalaze u .bag datoteci iz koje se naknadno spremaju u neki od formata pogodan za učitavanje oblaka točaka u specijalizirane softvere za tu namjenu (npr. CloudCompare) gdje se dalje može manipulirati podacima, mjeriti udaljenosti, uklapati (georeferencirati) oblake u prostor i slično. Tijek snimanja učionice br. 11 prikazan je na sljedećem kolažu slika (Slika 6.4.).



Slika 6.4. Faze snimanja Livox LiDAR-om

7. ZAKLJUČAK

Ključno i polazno pitanje s kojim je ovaj rad krenuo je iskoristivost i opravdanost niskobudžetnog nekomercijalnog SLAM sustava u svrhe geodetske izmjere, kartiranja i srodnih namjena. Dosadašnja iskustva sa komercijalnim SLAM sustavima, spremnim za korištenje, su vrlo dobra te su dobro prihvaćeni od strane struke. Jedini nedostatak takvih, a značajna prednost nekomercijalnih sustava je u bitno višoj cijeni. Sustavi poput ovoga većinom su još u eksperimentalnim fazama. Većina ljudi još uvijek je skeptična po pitanju korištenja SLAM sustava s LiDAR-om čvrstog stanja za kartiranje i izmjeru, te više vjeruju dosadašnjim „tradicionalnim“ načinima prikupljanja podataka o prostoru. Tome zasigurno pridonosi i nedovoljna informiranost o mogućnostima, a i slaba zastupljenost ove vrste LiDAR-a na našem tržištu.

Tehničke specifikacije korištenog Livox Mid-100 uređaja zadovoljavajuće su i ne zaostaju za skupljim modelima, te se s njim može dobiti dovoljno gust oblak točaka iz kojeg se lako mogu razabrati i sitniji detalji snimanog objekta ili površine. Međutim, kao i kod višestruko skupljih lasera, većina degradirajućih faktora dešava se uslijed kretanja. S obzirom na to, fokus treba postaviti na što bolju softversku podlogu i algoritme kojima se što više pogrešaka može minimalizirati ili, u najboljem slučaju, ukloniti. Tako bi se, primjerice, sustavu zasigurno moglo pomoći ukoliko bi se na dijelove trajektorije postavile fiksne „nepogrešne“ točke. Na taj bi se način smanjile pogreške uzrokovane duljinom trajektorije. Algoritam zatvaranja petlje također se pokazao kao dobar alat za uklanjanje tzv. „driftova“ i bolje spajanje snimaka različitih vremenskih epoha. Svakako bi za čim bolje rezultate kod snimanja većih površina trebalo težiti prema eliminiranju pokretnih objekata (ljudi, auti) koji dodatno zbunjuju sustav, otežavaju mu točno pozicioniranje i doprinose nastajanju šuma na rezultatima. Dubljom analizom svakog parametra koji je sadržan u datotekama algoritma i njihovom optimizacijom za različite zadatke može se prilagoditi način snimanja i udovoljiti specifičnim zahtjevima točnosti rezultata. Implementacija digitalne kamere, IMU jedinice i GNSS jedinice još je jedan način za dodatno poboljšanje rezultata, a i sam prikaz preklopljen sa digitalnim snimcima u boji, fotografijama, daje reprezentativniji i oku ugodniji prikaz snimljenog.

U ovom diplomskim radu uspješno se realizirao LOAM Livox algoritam pomoću Livox Mid-100 LiDAR-a čvrstog stanja, što daje pozitivan odgovor na pitanje oko iskoristivosti sustava u svrhu geodetskih izmjera i kartiranja. Boljim upoznavanjem i korištenjem gore spomenutih načina za poboljšavanje rezultata, a i razvijanjem novih, ova perspektivna metoda ima velik potencijal u kartiranju okoliša i geodetskim izmjerama te joj je predviđena svijetla budućnost.

LITERATURA

Berta, A., (2017): Procjena šumske biomase pomoću lidar tehnologije u degradiranim šumama bretuljkastoga i nizinskoga vegetacijskoga pojasa u središnjoj Hrvatskoj, Doktorski rad, Šumarski fakultet Sveučilišta u Zagrebu, Zagreb.

Bojić, M., Kogoj, D. (2017). Kalibracija terestričkih laserskih skenera. Geodetski glasnik, 51(48), 50-73.

Dehai, Z., Hao, G., Wei, S. (2012): Point cloud library PCL learning tutorial (ISBN 978-7-5124-0954-5), Beijing Aerospace Press 2012-10.

Durrant-Whyte, H., Bailey, T. (2006): Simultaneous Localization and Mapping: Part I, IEEE Robotics & Automation Magazine, 13(2), 99-110, doi: 10.1109/mra.2006.1638022.

Gikas, V. (2012): Three-Dimensional Laser Scanning for Geometry Documentation and Construction Management of Highway Tunnels during Excavation, article, Sensors, 2012, 12 (8), 11249-11270; doi: 10.3390/s120811249.

Kordić, B. (2014): Razvoj metode trodimenzionalnog terestričkog laserskog skeniranja kod određivanja i analize pomaka površine klizišta, doktorski rad, Geodetski fakultet, Zagreb.

Lasić, Z. (2008.): Primjena laserskih uređaja- Skripta, Geodetski fakultet Sveučilišta u Zagrebu, Zagreb.

Lemmens, M. (2011): Geo-information, svezak 5. serije Geotechnologies and the Enviroment, Delft University of Technology, Delft, Nizozemska.

Lentin, J. (2015): Mastering ROS for Robotics Programming (2015, Packt Publishing).

Lin, J., Zhang, F. (2019a): A fast, complete, point cloud based loop closure for LiDAR odometry and mapping.

Lin, J., Zhang, F. (2019b): Loam_livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV.

Lin, Y., Hyyppa, J., Jaakkola, A. (2014): Combining mobile and static terrestrial laser scanners to investigate individual crown attributes during foliation, Canadian Journal of Remote Sensing: Journal canadien de télédétection, 37:4, 359-375, doi: 10.5589/m11-045.

Matijević, H., Roić, M. (2002): Terestrički laserski skeneri, Geodetski list, 3, 171-187.

Miler M., Đapo A., Kordić B., Medved I. (2007): Terestrički laserski skeneri, Ekscentar, no. 10, pp. 35-38.

Nocerino, E., Menna, F., Remondino, F., Toschi, I., Rodríguez-Gonzálvez, P. (2017): Ispitivanje unutarnjih i vanjskih performansi dvaju prijenosnih mobilnih sustava za kartiranje, SPIE Digital library.

Ozisik, O., Yavuz, S. (2016): Simultaneous localization and mapping with limited sensing using extended Kalman filter and hough transform, original scientific paper, Technical Gazette, Vol. 23, No. 6, 1731-1738, doi: 10.17559/TV-20150830235942.

Pejić M. M. (2013): Tačnost modeliranja objekata tehnologijom terestričkog laserskog skeniranja, doktorska disertacija, Građevinski fakultet, Beograd.

Quigley, M., Gerkey, B., Smart, W.D. (2015) - Programming Robots with ROS (2015, O'Reilly Media).

Rusov, M. (2014): Terestrički laserski skener, seminarski rad, Fakultet tehničkih nauka, Novi Sad.

Staiger, R. (2009): Push the Button – or Does the „Art of Measurement" still Exist?, FIG Working Week, Surveyors's Key Role in Accelerated Development, Eilat, Isreal.

Stenz, U., Hartmann, J., Paffenholz, J. -A., Neumann, I. (2020): High-Precision 3D Object Capturing with Static and Kinematic Terrestrial Laser Scanning in Industrial Applications – Approaches of Quality Assessment, article, Remote Sensing 2020, 12, 290; doi: 10.3390/rs12020290.

Zogg, H.M. (2008): Investigations of High Precision Terrestrial Laser Scanning with Emphasis on the Development of a Robust Close-Range 3D-Laser Scanning System. Dissertation. Institute of Geodesy and Photogrammetry, ETH Zurich, Switzerland.

MREŽNE ADRESE

URL 1: Hrvatska enciklopedija, mrežno izdanje: lidar, https://enciklopedija.hr/natuknica.aspx?ID=36412, (25.2.2021.).

URL 2: Scielo, https://www.scielo.br/j/anaismp/a/D3WMPdSFfsbr9QdscQxW4vK/?lang=en, (25.02.2021.).

URL 3: ATS, https://ats.se/suppliers-faro-products.html, (05.05.2021.).

URL 4: LIDAR Magazine, https://lidarmag.com/2019/11/20/the-past-and-future-of-handheld-3d-scanning/, (05.05.2021.).

URL 5: LeddarTech, https://leddartech.com/video-3d-hybrid-flash-lidar/, (05.05.2021.).

URL 6: All About Circuits, https://www.allaboutcircuits.com/news/solid-state-lidar-faster-cheaper-better/, (05.05.2021.).

URL 7: Rutronik, https://www.rutronik.com/article/lidar-for-autonomous-driving-how-science-fiction-becomes-science-reality/, (26.10.2021.).

URL 8: All About Circuits, https://www.allaboutcircuits.com/news/solid-state-LiDAR-is-coming-to-an-autonomous-vehicle-near-you/, (26.10.2021.).

URL 9: Renishaw, https://www.renishaw.es/es/optical-encoders-and-lidar-scanning--39244, (26.10.2021.).

URL 10: Metrology, https://metrology.news/nikon-invests-in-3d-lidar/, (26.10.2021.).

URL 11: MDPI, https://www.mdpi.com/1424-8220/20/14/3964, (26.10.2021.).

URL 12: DJI Store, https://store.dji.com/hr/product/livox-mid-100-15pcs?vid=49211, (27.04.2021.).

URL 13: Livox, https://www.livoxtech.com/mid-40-and-mid-100/specs, (27.04.2021.).

URL 14: Livox Mid Series – User Manual, https://www.livoxtech.com/3296f540ecf5458a8829e01cf429798e/downloads/Livox%20Mid%20Series%20User%20Manual%20EN%2020190118.pdf, (27.4.2021.).

URL 15: LeddarTech, https://leddartech.com/technology-fundamentals/, (26.10.2021.).

URL 16: Mathworks, https://www.mathworks.com/discovery/slam.html, (20.10.2021.).

URL 17: Kudan, https://www.kudan.eu/kudan-news/an-introduction-to-slam/, (20.10.2021.).

URL 18: FME Community, https://docs.safe.com/fme/html/FME_Desktop_Documentation/FME_ReadersWriters/pcd/about_point_clouds.htm, (19.5.2021.).

URL 19: Asprs, https://www.asprs.org/divisions-committees/lidar-division/laser-las-file-format-exchange-activities, (19.05.2021.).

URL 20: FME Community, https://docs.safe.com/fme/html/FME_Desktop_Documentation /FME_ReadersWriters/pcd/pcd.htm, (11.10.2021.).

URL 21: PointCloudLibrary, https://pointclouds.org/documentation/tutorials/pcd_file_ format.html, (27.10.2021.).

URL 22: Develop PAPER, https://developpaper.com/python-converts-the-specified-point-cloud-file-asc-to-pcd-format/, (27.10.2021.).

URL 23: Paul Bourke, http://paulbourke.net/dataformats/ply/, (25.02.21.).

URL 24: Florida State University, https://people.sc.fsu.edu/~jburkardt/data/ply/ply.html,

(25.02.2021.).

URL 25: Linux, https://www.linux.com/what-is-linux/, (03.11.2021.).

URL 26: Ubuntu, https://ubuntu.com/community/debian, (03.11.2021.).

URL 27: Ubuntu, https://ubuntu.com/about, (03.11.2021.).

URL 28: DEV Community, https://dev.to/awwsmm/101-bash-commands-and-tips-for-beginners-to-experts-30je, (03.11.2021.).

URL 28: ROS Wiki, http://wiki.ros.org/ROS/Concepts, (06.05.2021.).

URL 29: Packt, https://subscription.packtpub.com/book/hardware_and_creative/ 9781788478953/1/ch01lvl1sec13/understanding-the-ros-filesystem-level, (06.05.2021.).

URL 30: Packt, https://subscription.packtpub.com/book/hardware_and_creative/ 9781788478953/1/ch01lvl1sec14/understanding-the-ros-computation-graph-level, (19.5.2021.).

URL 31: ROS Wiki, http://wiki.ros.org/Tools, (12.10.2021.).

URL 32: ROS Wiki, http://wiki.ros.org/rviz, (12.10.2021.).

URL 33: ROSWiki, http://wiki.ros.org/rosbag/Commandline, (12.10.2021.).

URL 34: Ubuntu, https://help.ubuntu.com/community/Repositories/Ubuntu, (14.5.2021.).

URL 35: ROS Wiki, http://wiki.ros.org/ROS/Tutorials, (23.5.2021.).

URL 36: CeresSolver, http://ceres-solver.org/, (23.05.2021.).

URL 37: Point Cloud Library https://pointclouds.org/about/#open, (27.4.2021.).

URL 38: GitHub, https://github.com/Livox-SDK/Livox-SDK, (15.12.2021.).

URL 39: GitHub, https://github.com/Livox-SDK/Livox-SDK/wiki/Livox-SDK-Communication-Protocol, (15.12.2021.).

URL 40: GitHub, https://github.com/Livox-SDK/livox_ros_driver, (15.12.2021.).

POPIS SLIKA

POPIS TABLICA

ŽIVOTOPIS

Nino Franković rođen je 15. lipnja 1995. godine u Rijeci. U Labinu završava Osnovnu školu Matije Vlačića i Srednju školu Mate Blažine, smjer Opća Gimnazija. Tijekom školskog, a kasnije i fakultetskog obrazovanja aktivno se bavi rukometom. Preddiplomski studij Geodezije i Geoinformatike na Geodetskom fakultetu Sveučilišta u Zagrebu upisuje 2014. godine, a uspješno ga završava 2018. stjecanjem akademskog naziva sveučilišni prvostupnik (baccalaureus) inženjer geodezije i geoinformatike (univ. bacc. ing. geod. et geoinf.). Godine 2018. upisuje diplomski studij Geodezije i geoinformatike, usmjerenje Geodezija na istoimenom fakultetu. Materinji mu je jezik hrvatski. Od stranih jezika aktivno govori engleski.