

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 56

**MODELSKI INFORMIRANO GLOBALNO
PLANIRANJE PUTANJE I UPRAVLJANJE
AUTONOMNOGA PLOVILA**

Enio Krizman

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 56

Pristupnik: **Enio Krizman (0069083848)**

Studij: Informacijska i komunikacijska tehnologija

Profil: Automatika i robotika

Mentor: doc. dr. sc. Đula Nađ

Zadatak: **Modelske informirane globalne planiranje putanje i upravljanje autonomnoga plovila**

Opis zadatka:

Potrebno je napraviti pregled literature iz područja modelski informiranih algoritama globalnog planiranja putanje za plovila, te implementirati algoritam globalnog planiranja putanje baziran na mapi cijena. Nadalje, treba implementirati algoritam generiranja trajektorije temeljen na kinematičkim ograničenjima modeliranoga plovila. Validaciju algoritama potrebno je provesti na nekoliko primjera u mapi Jadranskoga mora i to u ROS simulacijskom okruženju. Finalnu validaciju algoritama treba provesti u MARUS pomorskom simulatoru.

Rok za predaju rada: 28. lipnja 2024.

Ovaj je rad nastao na Zavodu za automatiku i računalno inženjerstvo na Fakultetu elektrotehnike i računarstva u Zagrebu pod vodstvom mentora doc. dr. sc. Đule Nađa i mentora dr. sc. Nadira Kapetanovića u sklopu znanstveno-istraživačkoga projekta. Ovim putem izražavam veliku zahvalnost mentorima na stručnom i izrazito inspirativnom vođenju cijelog procesa izrade ovoga diplomskoga rada. Zahvaljujem se na svakome savjetu, preporuci, smjernici, uputama i odgovorima bez kojih ne bih uspio odraditi zadatok Modelske informirano globalno planiranje putanje i upravljanje autonomnoga plovila te što su mi omogućili samostalan pristup zadanoj temi. Zahvalu upućujem i svim svojim profesorima koji su nas podučavali svemu onome što je prethodilo današnjim znanstvenim dostignućima kao i onome što slijedi u budućnosti, tražeći od nas da uvijek budemo kritički i misaono spremni suočiti se sa svim životnim izazovima kako u poslovnom tako i u privatnome životu. Ovih će mi pet proteklih godina studiranja ostati u dubokom sjećanju kao zahtjevne i uz puno odricanja, ali ništa manje lijepo i uz mnoštvo uspomena. Zahvalan sam i ponosan što sam bio dijelom tima studenata i profesora Fakulteta za elektrotehniku i računarstvo u Zagrebu. Također, izražavam zahvalu i svim kolegama i prijateljima koji su bili uz mene u onim lijepim, ali i zahtjevnim trenutcima na fakultetu. Isto tako hvala i mojoj obitelji koja je uvijek bili uz mene i bez čije podrške i poticaja ne bih imao priliku doći do završetka studija. Hvala i svima koji su na bilo koji način utjecali i pomogli pri izradi ovoga rada.

Sadržaj

1. Uvod	4
1.1. Objašnjenje pojmove modelski informiranog globalnog planiranja putanje	4
1.2. Postavljanje hipoteze rada	6
1.3. Algoritmi globalnog planiranja putanje	8
1.4. Struktura rada i argumentacija odabira ROS2 okruženja za razvoj programske potpore	11
2. Modelske informirane globalne planiranje putanje plovila	13
2.1. Arhitektura ROS2 programske potpore za planiranje putanje plovila	14
2.1.1. ROS2 radni prostor	14
2.1.2. ROS2 paketi	15
2.1.3. ROS2 mehanizmi komunikacije i koncept crne kutije	16
2.2. Objašnjenje i pretvorbe između globalnog i lokalnog koordinatnog sustava	19
2.3. Izrada pomorske mape geografskih koordinata	26
2.3.1. Struktura i pokretanje ROS2 paketa map_maker	26
2.3.2. Preuzimanje i obrada podataka s OpenStreetMap internetske stranice	30
2.3.3. Pronalaženje koordinata obale obradom fotografije i postavljanje zona udaljenosti	32
2.3.4. Izrada kolaž mape geografskih koordinata postupkom poravnjanja koordinata i spremanje rezultata	38
2.3.5. Objavljivanje geografskih koordinata mape unutar ROS2 okvira	41

2.4. Postavljanje polazišne i odredišne točke planiranja putanje plovila i vizualizacija podataka	42
2.4.1. Struktura i pokretanje ROS2 paketa path_planning_client	42
2.4.2. Vizualizacija podataka u RViz2 vizualizacijskom alatu	47
2.4.3. Postavljanje polazišne i odredišne točke te pokretanje ROS2 action client i action server za planiranje putanje plovila	51
2.5. Globalno planiranje putanje plovila bazirano na mapi cijena prostora i process interpolacije putanje	58
2.5.1. Struktura i pokretanje ROS2 paketa path_planning_server	59
2.5.2. Testiranje algoritama globalnog planiranja putanje	62
2.5.3. Postupak postavljanja mape cijena i prilagodba D* lite algoritma za modelski informirano globalno planiranje putanje plovila	68
2.5.4. Interpolacija putanje postupcima raspoređivanja točaka i prilagođavanja krivulja	78
2.6. Postavljanje polazišne i odredišne točke planiranja putanje plovila i vizualizacija podataka	79
2.6.1. Struktura i pokretanje ROS2 paketa path_planning_client	80
2.6.2. Vizualizacija podataka u RViz2 vizualizacijskom alatu	84
2.6.3. Postavljanje polazišne i odredišne točke te pokretanje ROS2 action client i action server za planiranje putanje plovila	89
2.6.4. Objavljivanje rezultata putem putem action client i action server mehanizma komunikacije i prikaz rezultata u RViz2 vizualizacijskom alatu	100
2.6.5. Rezultati i karakteristike konačne putanje u ovisnosti o parametrima procesa planiranja putanje	101
3. Zaključak	109
Literatura	112
Sažetak	113
Abstract	114

A: Programska potpora (the code)	115
---	------------

1. Uvod

1.1. Objašnjenje pojma modelski informiranog globalnog planiranja putanje

Modelska informirana globalna planiranja plovila (engl. Model-based path planning) se odnosi na metode planiranja putanja koje koriste model okruženja i dinamike plovila kako bi odredile optimalnu i efikasnu globalnu putanju plovila po različitim kriterijima. Ove se metode oslanjaju na matematičke modele okoline u kojoj se planiraju putanje plovila te matematičke i fizikalne modele izvedivosti putanje plovila s obzirom na njegove kinematičke i dinamičke karakteristike. Prednosti su modelski informiranog planiranja putanje preciznost zbog korištenja modela, optimizacija putanje po različitim kriterijima (duljina i vrijeme putovanja, ušteda energije, namjena...) i robusnost rukovanja u složenim dinamičkim okruženjima. Globalno (offline) planiranje putanje odnosi se na proces određivanja optimalne putanje od polazišne do odredišne točke putanje u poznatom okruženju, bez zahtjeva za radom u realnom vremenu. Odnosno, navedena globalna putanja se planira prije početka plovidbe na osnovi unaprijed poznatih informacija o prostoru očitanih s geografske mape plovnog područja.

Matematički model okruženja uključuje geografsku mapu plovnog područja s istaknutim koordinatama obale koje naznačuju prepreku, koordinatama udaljenosti od obale, dubinama mora te prerekama na moru u što se ubrajaju manji otoci, hridi i signalizacijski objekti. Također, u matematički se model ubrajaju koordinate polazišne i odredišne točke putanje kada su postavljene. Na osnovi se geografske mape plovnog područja stvara mapa cijena prostora koja predstavlja okolinu u obliku rešetke (eng. grid) gdje svaka komponenta ima pridruženu cijenu koja označava trošak prolaska kroz tu komponentu. Prostor se dijeli na kvadratne (ili ponekad heksagonalne ili pravokutne) kompo-

nente koje pojednostavljaju modeliranje okruženja i planiranje putanje čime se zapravo prostor pretvara u diskretni skup elemenata. Glavne prednosti reprezentacije prostora kao diskretnog skupa elemenata jesu mogućnost brzog i učinkovitog pretraživanja elemenata različitim računskim postupcima u širem spektru algoritama kao i jednostavna otkrivanja promjena između susjednih elemenata s obzirom na jasno definirane grane područja. Kod reprezentacije plovnog područja u obliku rešetke, svaka komponenta predstavlja određenu geografsku koordinatu stvarnog prostora s pripadajućom cijenom. Postavljene cijene komponenata diskretnog skupa elemenata mogu biti temeljene na različitim kriterijima kao što su udaljenost od obale, dubina mora ili drugi faktori koji utječu na sigurnost i efikasnost kretanja plovila. Uloga je mape cijena dakle generiranje optimalne putanje plovila u smislu udaljenosti, vremena trajanja plovidbe ili potrošnje energije uz ispunjenje kriterija sigurnosti plovidbe. Kriteriji su sigurne plovidbe osiguranje dovoljne udaljenosti isplanirane putanje od obale, plitkih područja ili prepreka na moru te prolazak sredinom kanala u uskim morskim tjesnacima. Zaključno, matematički model okruženja modelski informiranog globalnog planiranja putanje predstavlja diskretizirani skup elemenata raspoređenih u obliku rešetke (mapa cijena prostora) gdje je svaka komponenta reprezentacija određene geografske koordinate, a cijena prolaska kroz komponentu je određena s obzirom na različite kriterije sigurne plovidbe i optimalnosti putanje.

Matematički i fizikalni modeli dinamike plovila podrazumijevaju uključenje izvedivosti putanje s obzirom na tip, dinamičke i kinematičke karakteristike plovila u mapu cijena prostora. U ovisnosti o kriteriju izvedivost se putanje u svakom koraku procesa planiranja putanje plovila ažurira cijena prolaska kroz komponentu rešetke. Rezultat uključenja dinamike plovila u proces planiranja putanje koji koristi mapu cijena prostora kao glavni alat planiranja jest smanjenje ukupnog broja komponenta i optimizacija algoritma. Naime, od polazišne do odredišne točke u prostoru se nalaze slijedne komponente rešetke kojima plovila s obzirom na njihov dinamički model mogu proći. Odnosno, u diskretnom skupu podataka, komponente rešetke koje su dijelom izvedivih putanja imaju konačne cijene, a cijena ostalih komponenata je postavljena na vrijednost beskonačno. Zahtjev izrade algoritma planiranja putanje u ovome je radu bio razviti univerzalni algoritam planiranja koji će funkcionirati za veliku većinu manjih i srednjih plovila. Razlog tome je što algoritam nije izrađen za specifičnu vrstu i namjenu plovila, već u svrhu is-

pitivanja koncepata globalnog planiranja putanje. Sukladno navedenom, algoritam ne uključuje dinamički model određenog plovila dalje od prilagodbe koraka pretrage algoritma uzorkovanog s obzirom na duljinu plovila. Nadaje, radi nedostupnosti podataka o dubinama mora, iste nije bilo moguće uključiti u izradu mape cijena prostora. Međutim, informacije o dubini mora i dinamičkim karakteristikama plovila su zasigurno veoma značajne stavke koje je potrebno uključiti u navedene modele globalnog planiranja putanje prije nego li rezultati ovoga rada budu testirani u stvarnom okruženju.

1.2. Postavljanje hipoteze rada

Hipoteza koju postavlja ovaj rad je da se metode planiranja putanje plovila koja će biti predstavljene mogu primijeniti za planiranje putanje stvarnih plovila srednjih i manjih duljina bez uključenja dinamičkog modela u trenutku kada budu poznate informacije matematičkog modela okruženja o dubinama i statičnim preprekama na moru kao što hridi, manji otoci ili postavljenih objekata za signalizaciju. Navedena hipoteza je postavljena s premisom da metode planiranja putanje plovila izvršavaju zadaću planiranja plovila od polazišne do odredišne točke bez specifičnih namjena. Pod pojmom specifične namjene pripadaju pretraga područja po određenom uzorku, testiranje performansi plovila poput Zig-Zag manevra ili bilo koja druga namjena koja izlazi izvan okvira osnovne zadaće algoritma planiranja. Osnovna je zadaća svakog algoritma planiranja putanje neovisno o modelu okruženja dovesti objekt za koji se putanja planira najkraćom sigurnom rutom od polazišne do odredišne točke putanje.

Prepostavka hipoteze da dinamički model plovila nije neophodan za planiranje sigurne putanje proizlazi iz činjenice da su plovna područja u manjoj mjeri dinamički zahtjevna i višestruko prostorno veća od dinamičkih područja u kojima putanje planiraju mobilni roboti, autonomni automobili i letjelice. Pojam manje zahtjevnog dinamičkog okruženje podrazumijeva da planiranje putanje plovila ne postavlja zahtjev na brze i nagle promjene smjera zbog samih geografski obilježja prostora. Također, uz dovoljno informacija o okruženju moguće je isplanirati optimalnu putanju koja ne predstavlja sigurnosni problem jer putanja prolazi na sigurnim udaljenostima od obale, plitkih područja i statičnih prepreka. Ranije je spomenuto da se globalno planiranje putanje vrši prije početka plovidbe te ne posjeduje informacije o dinamičkim i statičkim preprekama

na moru u trenutku plovidbe. Shodno navedenom, sigurna je udaljenost bitan čimbenik globalnog planiranja putanje jer osigurava veće vrijeme reakcije pri plovidbi na čimbenike koje je nemoguće predvidjeti prilikom planiranja putanje neovisno izvršava li plovidbu autonomni sustav ili kapetan plovila. Također, vrijeme reakcije u plovnom području zbog njegovog opsega znatno je veće od vremena reakcije u dinamički zahtjevnijim okruženjima. Osiguranim većim vremenom reakcije pri plovidbi smanjuje se vjerljivost od nasukavanja o obalu ili plitka područja te sudaranja o prepreke prilikom promjene smjera plovidbe od definirane globalne putanje. Veće vrijeme reakcije također je jedan od bitnijih čimbenika kod prilagodbe plovidbe na utjecaj vjetra i morskih struja. Reakcije na navedene čimbenike koje nisu poznati globalnom algoritmu planiranja putanje nalaze se u znanstvena područja lokalnog planiranja putanje i upravljanja plovilima, što nije predmet ovoga rada. Međutim, svakako ih je važno spomenuti radi sagledavanja šire slike cjelokupnog procesa planiranja putanje plovila u dinamičkim okruženjima pod što spada globalno i lokalno planiranje putanje plovila.

Lokalno se planiranje putanje plovila odnosi na proces generiranja i prilagođavanja putanje za plovilo u kratkom vremenskom okviru, koristeći senzorske podatke za izbjegavanje prepreka i navigaciju kroz dinamičko okruženje. Najbolji dostupni algoritam lokalnog planiranja putanje je i dalje svijest osobe koja pomoću iskustva, procjena i intuicije uspješno upravlja plovilom kroz dinamička okruženja. Na čovjekovu svijest utječu različiti faktori kao što su razina iskustva upravljanja plovilom, poznavanje ili nepoznavanje planirane putanje, otpornost na stres, humor i slično. Upravo se stoga posljednjih godina u različitim industrijskim nastojanjima razviti sučelje koje pomoću senzorskih podataka stvara situacijsku percepciju prostora što uvelike osobi pomaže prilikom upravljanja plovila u dinamičkim okruženjima.

Zaključno, neovisno upravlja li plovilom osoba ili autonomni sustav vrijeme je reakcije značajan čimbenik upravljanja plovila u dinamičkim okruženjima. Uz pravilno postavljenu reprezentaciju geografskog prostora koja slijedi pravila sigurnosti putanje moguće je generirati izvedivu putanju za srednja i mala plovila bez poznavanja dinamičkog modela. Pretpostavka slijedi iz činjenice da je vrijeme reakcije na nepredviđene čimbenike u plovnom području relativno veliko, što osigurava da se neovisno o dinamičkim karakteristikama plovila, mogu izbjegći sigurnosni problemi u dinamičkim okruženjima.

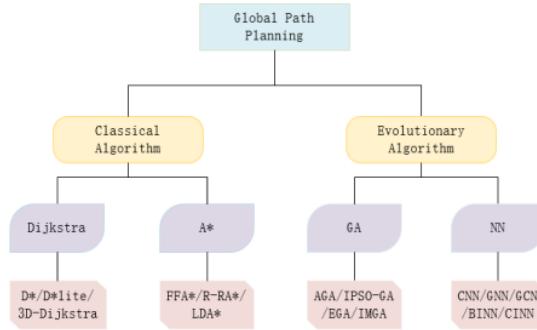
S druge strane, u dinamički zahtjevnijim područjima su potrebne brze reakcije i promjene smjera, kako bi se izbjegli sigurnosni problemi koje postavlja okruženje te je u većini takvih okruženja dinamički model objekta za koji se putanja planira neophodan. Pod pojmom dinamički zahtjevnijih područja ubrajaju se područja na kopnu i zraku u kojima je vrijeme reakcije smanjeno. U sigurnosne probleme pripadaju sudaranje o prepreke koje su na relativno malim udaljenostima te ih je brojčano znatno više. Također, postoji mogućnost zaglavljenja algoritma planiranja putanje (eng. dead lock) ukoliko se algoritam nađe u području pretrage gdje nisu dostupni sljedeći koraci s obzirom na dinamičke modele. Shodno ranije navedenom, planiranje putanje plovila na moru je bitno drugačije od planiranja putanje vozila na kopnu i letjelica u zraku zbog drugačijeg dinamičkog okruženja i fizikalnih zakona koja u istima vladaju te dinamičkih karakteristika vozila, letjelica i plovila. Nadalje, u složenijim dinamičkim okruženjima veći je naglasak na lokalnom planiranju putanje koje zahtjeva poznavanje dinamičkog modela predikcije kretanja te stvaranje situacijske svijesti o okruženju.

Prethodno navedene činjenice navode na zaključak da karakteristike planiranja putanje u područjima složenijih dinamičkih karakteristika nemaju mnogo toga zajedničkog s karakteristikama planiranja putanje na moru zbog različitog vremena reakcije i opsega područja. Najbitniji čimbenici planiranja putanje jesu unaprijed poznate statičke karakteristike i nepoznate dinamičke karakteristike okruženja te iako su algoritmi bazirani na sličnim principima, njihova primjena i uporaba bitno je drugačija u različitim okruženjima te se zato ne može govoriti o univerzalnosti procesa planiranja putanje. Odnosno, ako neka pravila i premise planiranja putanje vrijede u jednom okruženju, ne znači nužno da su primjenjiva u drugome okruženju.

1.3. Algoritmi globalnog planiranja putanje

Po definiciji globalno planiranje puta je opsežna izvanmrežna (offline) metoda planiranja optimalne putanje koja se temelji na pruženim informacijama o plovidbenom području putem geografskih mapa te ne zahtijeva rad u realnom vremenu [1]. Algoritam globalnog planiranja putanje prikuplja informacije o okolišu modelirajući okruženje uz zadane zahtjeve te objavljuje plan kretanja plovila određenom putanjom. Zahtjevi mogu biti različiti u ovisnosti o zadaći i karakteristikama plovila. Algoritmi globalnog planira-

nja putanje nemaju uvid u informacije lokalnog tipa kao što su kretanje plovila i ostalih prepreke na moru te shodno navedenom nisu u potpunosti optimalna za rad u realnom vremenu, odnosno tijekom plovidbe. Naime, globalna putanja se planira prije početka plovidbe na osnovi unaprijed poznatih informacija o prostoru očitanih s geografske mape plovnog područja. U konačnici, zadaća je algoritma generirati održiv put od polazišne do odredišne točke putanje u poznatom plovidbenom prostoru. Trenutačno su glavne metode globalnog planiranja putanje u pomorskoj industriji inačice algoritama nastalih iz tradicionalnih Dijkstra i A start (*) algoritama planiranja putanje te novo razvijeni algoritmi u što se ubrajaju genetski algoritmi i algoritmi neuronskih mreža, što je prikazano na slici 1.1.



Slika 1.1. Grafički prikaz stabla algoritama globalnog planiranja putanje. Preuzeto s [1]

Dijkstra algoritam je klasičan algoritam za pretraživanje najkraćih putanja. Algoritam je formulirao E.W. Dijkstra 1959. godine. Iako je učinkovit za pronalaženje najkraće putanje do svih čvorova u grafu, njegova računska složenost može biti problematična. Kako bi se povećala učinkovitost, razvijene su različite varijante algoritma. Jedna od varijanti dodaje ključne čvorove i dijeli regije kako bi se smanjilo vrijeme obrade, dok druga smanjuje izračun nekritičnih čvorova. Hjernarhijski Dijkstra algoritam spremi pretražene lokacije kako bi izbjegao ponovljene pretrage. Postoji i trodimenzionalna varijanta koja preciznije pronalazi globalno optimalne putanje. D*Lite algoritam i njegove poboljšane verzije su prilagođene za dinamička okruženja, predviđajući promjene i poboljšavajući učinkovitost izbjegavanjem dvostrukih izračuna.

Algoritam A* je predložen 1967. godine i koristi heurističku pretragu za pronalaženje najkraće rute između dva čvora. Brži je od Dijkstra algoritma i osigurava putanju s najnižim troškom, ali se oslanja na heurističke funkcije koje mogu smanjiti glatkoću

putanje, ako su složene ili neefikasne. Poboljšanja A* algoritma uključuju proširenje broja susjednih točaka za bolju glatkoću putanje i optimizaciju heurističke funkcije za smanjenje vremena računanja. Varijante poput FFA*, LDA*, i R-RA* uvode sigurnosne parametre i optimizacije koje poboljšavaju glatkoću i sigurnost putanje, ali mogu povećati vrijeme računanja ili uopće ne pronaći globalno optimalne putanje.

Genetski algoritmi (GA) koje je predložio John Holland, koriste princip genetike i prirodne selekcije te su široko primjenjivi za rješavanje složenih problema pretraživanja. Međutim, klasični genetski algoritmi mogu imati nisku konvergenciju i biti izrazito spori prilikom računanja u dinamičkim nepoznatim okruženjima zbog ovisnosti o funkciji optimizacije. Unapređenja genetskih algoritama usmjerena su na poboljšanje optimizacije genetskih parametara kako bi se povećala učinkovitost i brzina konvergencije algoritma.

Neuronske mreže (NN) su uveli McCulloch i Pitts 1944. godine i koriste se za navigaciju u nepoznatom okruženju. Metode temeljene na neuronskim mrežama, kao što su konvolucijske neuralne mreže (CNN) i grafne konvolucijske mreže (GCN) poboljšavaju planiranje putanja prikupljanjem različitih informacija. Iako se ove metode mogu suočiti s izazovima u stvarnom vremenu i velikim skalama podataka, unapređenja kao što su rezidualne konvolucijske mreže (R-CNN) i bio-inspirirane neuralne mreže (BINN) nude poboljšanja u točnosti i glatkoći putanja, smanjujući vrijeme obrade.

Zaključno, globalno planiranje putanje predstavlja ključnu komponentu u pomorskoj navigaciji, omogućujući plovilima pronalazak optimalne putanje koristeći unaprijeđene informacije iz geografskih mapa. Unatoč razvoju naprednih algoritama, tradicionalni algoritmi ostaju glavni izbor u pomorskoj industriji zbog svoje pouzdanosti i učinkovitosti. Iako inovativni, genetski algoritmi i algoritmi neuronskih često imaju probleme niske konvergencije i velike računske složenosti. Također su spori i manje pouzdani u dinamičkim i nepoznatim okruženjima prilikom izvanmrežnog planiranja putanje. Zbog ranije navedenog, za proces planiranja putanje u ovome su radu odabrani klasični algoritmi globalnog planiranja putanje. Karakteristike nekoliko inačica klasičnih algoritama ispitne su u poglavlju 2.5. te je na osnovi dobivenih rezultata odlučeno koji će od algoritama biti odabran kao okosnica procesa izrade modelski informiranog globalnog planiranja putanje.

1.4. Struktura rada i argumentacija odabira ROS2 okruženja za razvoj programske potpore

Planiranje se putanje plovila odnosi na proces određivanja optimalne rute pomoću različitih tehnika i algoritama od točke polazišta do točke odredišta na geografskoj mapi prostora. Postupak izrade programske potpore za globalno planiranje putanje plovila je složeni proces podijeljen u tri logičke cjeline, odnosno tri cjeline programske potpore. Shodno definiciji, prva logička cjelina opisana u poglavlju 2.3. je izrada mape geografskog prostora s označenim hidrografskim obilježjima. Pomorske mape, koje se koriste za planiranje putanje i navigaciju plovila, su geografske mape plovidbenog područja, konstruirane u usvojenim kartografskim projekcijama. One sadrže sve potrebne elemente za orientaciju i sigurnu navigaciju, uključujući morfologiju morskog dna, koordinate obale, udaljenosti od obale te područja opasnosti. Druga logička cjelina opisana u poglavlju 2.6. predstavlja postupak postavljanja polazišne i odredišne točke pomoću različitih vizualizacijskih i programskih tehnika na izrađenoj mapi geografskog prostora. Navedene tehnike jesu postavljanje točaka u trodimenzionalnom vizualizacijskom okruženju ili putem programa unosom geografskih koordinata prostora. Treća logička cjelina opisana u poglavlju 2.5. je izrada algoritma modelski informiranog globalnog planiranja putanje. Navedeni algoritam koristi jedan od algoritama klasičnog planiranja putanje zajedno s tehnikama interpolacije dobivene putanje za dobivanje optimalne, glatke i izvedive putanje plovila. Kao ulazni se podatci algoritma koriste dobivena geografska mapa prostora te polazišna i odredišna točka što kazuje da su zapravo navedene cjeline poredane kronološki s obzirom na njihovu ulogu i vrijeme izvršavanja u cjelokupnom procesu.

Navedene logičke cjeline izvršavaju različite zadaće u procesu planiranja putanje plovila. Također, zadaće se izvršavaju u ovisnosti jedna o drugoj pa su stoga potrebni različiti mehanizmi komunikacije i arhitekture organizacije programske potpore. Također, potrebno je osigurati da ažuriranjem ili promjenom bilo koje od programskih cjelina procesa, ostale cjeline neometano funkcioniraju te s novom ili ažuriranom cjelinom razmjenjuju informacije na jednak način kao i s prethodnom verzijom. Primjerice, ukoliko se promijeni programska cjelina za izradu mape geografskog prostora, programske cjeline postavljanja polazišne i odredišne točke i primjene algoritma planiranja putanje tre-

bale bi zadržati jednake funkcionalnosti. Nadalje, u procesu planiranja putanje potrebno je praćenje međurezultata procesa u realnom vremenu kako bi se ispravile pogreške u procesu optimizacije algoritma. U narednim je fazama projekta planirano testiranje programske potpore u stvarnome okruženju pomoću senzora za praćenje pozicije plovila što također treba uzeti u obzir u procesu izrade programske potpore.

Zbog zahtjeva za modularnošću i fleksibilnošću programske potpore i komunikacijskih alata, ROS2 je logičan odabir za razvoj sustava za planiranje putanje plovila. ROS2 se u dalnjem tekstu naziva ROS2 okvirom kako bi se razlikovao od pojma ROS2 paketa koji čine specifične komponente unutar tog okvira. Modularna arhitektura ROS2 okvira omogućava razdvajanje sustava na neovisne programske cjeline koje mogu izvršavati specifične zadatke samostalno. ROS2 također nudi različite mehanizme komunikacije temeljene na pouzdanoj tehnologiji distribuiranog upravljanja podacima poznatoj kao DDS (Data Distribution Service). Dodatno, ROS2 podržava integraciju različitih senzora koji prate poziciju plovila i druge relevantne podatke u stvarnom vremenu. Bitno je napomenuti da ROS2 okvir podržava rad i u distribuiranim sustavima, što znači da se različite komponente mogu pokrenuti na različitim računalima ili uređajima. Trenutno, najnovija inačica ROS2 okvira, nazvana Iron Irwin, korištena je u procesu razvoja programske potpore.

Glavni dio rada 2. opisuje arhitekturu programske potpore za globalno planiranje putanje podijeljene u nekoliko cjelina. U prvoj poglaviji predstavljene tehničke pojedinosti organizacije programske potpore i alata komunikacije u ROS2 okviru. Drugo poglavje detaljno opisuje koncept izrade lokalnog koordinatnog sustava pomoću informacija o geografskim koordinatama pojedinog prostora s pripadajućim funkcijama pretvorbe koordinatnih sustava. Navedeni je koncept veoma bitan za razumijevanje ostatka rada jer ga koriste sva naredna poglavija. Naposljetku, kao što je ranije opisano posljednja tri poglavija zapravo predstavljaju tri logičke cjeline izrade modelski informiranog globalnog planiranja putanje. Svaka logička cjelina ujedno je i ROS2 paket te će karakteristike i arhitektura komunikacije paketa također biti detaljno razjašnjeni. Ideja glavnog dijela rada je pružiti detaljan uvid u proces modelski informiranog globalnog putanje, a također može poslužiti i kao tehnička dokumentacija programske potpore budućim korisnicima sustava.

2. Modelska informirana globalno planiranje putanje plovila

Proces je izrade funkcionalne programske potpore za planiranje putanje plovila veoma kompleksan i sastoji se od nekoliko logičkih, odnosno programskih cjelina organiziranih pomoću ROS2 okvira. Kroz glavni dio ovoga rada, odnosno kroz ovo poglavlje bit će detaljno opisana arhitektura sustava i sve njezine sastavnice koje čine modelski informirano globalno planiranje putanje. U uvodnom je poglavlju 2.1. opisana arhitektura ROS2 radnog prostora i pripadajućih paketa te mehanizama komunikacije korištenih za planiranje putanje plovila. Nadalje, u drugome je poglavlju 2.2. predstavljen koncept izrade lokalnog koordinatnog sustava na osnovi geografskih koordinata globalnog koordinatnog sustava. Logičke cjeline planiranja putanje plovila predstavljene su ROS2 paketima pa shodno navedenom naredna poglavlja opisuju strukturu pripadajućih paketa te njihovu ulogu u procesu izrade modelski informiranog globalnog planiranja putanje. U poglavlju 2.3. je predstavljen koncept izrade mape geografskog prostora u kojem će se odvijati planiranje putanje plovila. Nadalje, poglavlje 2.6. opisuje postavljanje polazišne i odredišne točke putanje korištenjem računalnih i vizualizacijskih alata. Posljednje poglavlje 2.5. opisuje izradu algoritma za planiranje putanje plovila korištenjem D* lite algoritma i tehnika interpolacije putanje. Također, u posljednjem su poglavlju predstavljeni rezultati procesa izrade modelski informiranog globalnog planiranja putanje.

Glavni dio ovoga rada također može poslužiti budućim korisnicima programske potpore kao tehnička dokumentacija jer sadrži sve bitne informacije o konceptima, arhitekturi i naredbama za pokretanje sustava modelski informiranog globalnog planiranja putanje plovila.

2.1. Arhitektura ROS2 programske potpore za planiranje putanje plovila

Robot Operating System 2 (ROS2) je napredni međuprogramska paket (okvir) za razvoj programske potpore koji služi kao industrijski standard za programiranje robotskih sustava, a razvijen je kao nasljednik originalnog ROS međuprogramskog paketa. ROS2 je dizajniran kako bi unaprijedio stabilnost, sigurnost i pouzdanost u usporedbi s prethodnom verzijom. Omogućuje inženjerima i istraživačima lakše razvijanje sofisticiranih robotskih aplikacija integrirajući kompleksne sustave senzora, planiranja putanja, upravljanja pokretima i komunikacije između različitih dijelova robotskog sustava. Glavne značajke ROS2 uključuju podršku za različite platforme (Windows, Linux, macOS), podršku za različite programerske jezike (C++, Python), modularnu arhitekturu koja olakšava integraciju i prilagodbu, kao i napredne alate za dijagnostiku i simulaciju. Također, ROS2 nudi mogućnost vizualizacije podataka putem RViz2 vizualizacijskog alata. Glavne su organizacijske cjeline ROS2 okvira radni prostor i paketi koji koriste različite mehanizme komunikacije. Uloga ROS2 radnog prostora, paketa i mehanizama komunikacije u procesu izrade programske potpore za planiranje putanje plovila bit će objašnjena u ovome poglavlju.

2.1.1. ROS2 radni prostor

Proces izrade modelski informiranog planiranja putanje plovila se sastoji od nekoliko koraka raspoređenih u programske cjeline s obzirom na njihovu ulogu. Navedene programske cjeline zapravo predstavljaju ROS2 pakete unutar ROS2 radnog prostora (eng. workspace) imena *path_planning_ws* kao vrhovne cjeline organizacije programske potpore. Radni prostor u ROS2 okviru paketu je ključna organizacijska struktura koja podržava razvoj, testiranje i implementaciju različitih aplikacija, omogućujući učinkovito upravljanje procesom i resursima projekta. Struktura radnog prostora je standardizirana te se sastoji od nekoliko programskih mapa. Prva mapa je build koja se koristi tijekom za pohranu privremenih datoteka i objektnih datoteka koje su potrebne za izgradnju paketa. u install mapi se instaliraju gotove izvršne datoteke, konfiguracijske datoteke i ostali resursi nakon što su paketi uspješno izgrađeni. Ovo je krajnji rezultat procesa izgradnje, gdje se paketi instaliraju za kasniju upotrebu ili distribuciju. Mapa log sadrži

log datoteke koje se generiraju tijekom procesa izgradnje ili rada ROS 2 čvorova. Ove datoteke mogu biti korisne za dijagnosticiranje problema ili analizu ponašanja aplikacije. Međutim, najznačajnija programska mapa s pohranjenim ROS2 paketima cjelokupnog projekta je mapa src. Pojam izgradnje radnog prostora podrazumijeva izradu programskih mapa build, install i log na osnovi konfiguracija paketa iz src mape te se pokreće unutar Linux terminala pomoću naredbe *colocn build*. Izgradnja radnog prostora omogućuje izravan pristup izvršnim datotekama iz paketa install te njihovo pokretanje unutar ROS2 okvira. Radni prostor *path_planning_ws* karakteriziraju paketi *map_maker*, *path_planning_client*, *path_planning_server* i *user_action_interfaces* iz src programske mape radnog prostora te će njihova uloga biti objašnjena u narednom poglavlju.

2.1.2. ROS2 paketi

ROS2 paketi su organizirani skupovi datoteka koje zajedno čine funkcionalne jedinice programske potpore. Svaki ROS2 paket definira svoju funkcionalnost, ovisnosti o drugim paketima te način na koji se integrira s ostatkom sustava. Dvije ključne datoteke u svakom ROS2 paketu bazirane na Python programskom jeziku su package.xml i setup.py. Datoteka package.xml je XML datoteka koja opisuje osnovne informacije o paketu, uključujući njegovo ime, verziju, autora, ovisnosti o drugim paketima te druge metapodatke. Ova datoteka je bitna za sustav upravljanja paketima u ROS2 kako bi okvir mogao pravilno instalirati, upravljati i rukovati paketima unutar distribucije. S druge strane, setup.py je Python skripta koja se koristi za konfiguraciju i instalaciju paketa izvan ROS2 okvira. Ova skripta definira kako će se paket instalirati, koje Python module treba uključiti te druge specifične postavke potrebne za integraciju paketa.

Kao što je prethodno navedeno u uvodu poglavlja, paketi se koriste za organizaciju logičkih cjelina postupka izrade modelski informiranog planiranja putanje plovila. Shodno navedenom, prvi paket *map_maker* ima ključnu ulogu u procesu izrade geografskih mapa prostora i njihovog integriranja u ROS2 okvir putem ROS2 izdavača te je detaljnije opisan u poglavlju 2.3. Drugi paket je *path_planning_client* koji je zadužen za vizualizaciju podataka unutar RViz2 vizualizacijskog alata, upravljanje procesom postavljanja polazišne i odredišne točke te objavljivanje istih unutar ROS2 okvira putem action client i action server mehanizma komunikacije za planiranje putanje plovila. Rviz2 vizualizacijski alat se koristi i za vizualizaciju geografskog prostora, vizualizaciju i postavljanje polazišne i

odredišne točke putanje te vizualizaciju konačne putanje plovila. Pretvorba podataka u oblik pogodan za vizualizaciju također je zadaća spomenutog paketa, a navedeni je paket detaljnije prikazan u poglavlju 2.6. Uloga trećeg paketa *path_planning_server* je uspostava action client i action server komunikacije s *path_planning_client* paketom putem koje se razmjenjuju informacije o polazišnoj i odredišnoj točki putanje, području pretrage algoritma i konačnoj putanji u interpoliranom i neinterpoliranom obliku. Zadaća navedenog paketa je putem algoritma i tehnika interpolacije putanje generirati sigurnu, glatku i izvedivu putanju plovila. Jedini paket koji nije opisan u zasebnom poglavlju, a ima važnu ulogu u procesu planiranja putanje je paket *user_action_interfaces*. Naime, paket se koristi kao svojevrsni spremnik za prilagođene poruke koje drugi paketi koriste tijekom komunikacije između pripadajućih čvorova ili prilagođene akcije koje koristi akcijski klijenti i serveri. Nadalje, ostali paketi u svojoj package.xml konfiguracijskoj datoteci inicijaliziraju *user_action_interfaces* paket te imaju pristup sadržaju njegovih poruka i akcija. Imena i sadržaj prilagođenih poruka i akcija bit će detaljnije predstavljen kroz naredna poglavlja, ali važno je naglasiti kako paket njihovog porijekla nije dijelom standardnog ROS2 okvira već je dijelom *path_planning_ws* radnog prostora, odnosno *user_action_interfaces* paketa.

2.1.3. ROS2 mehanizmi komunikacije i koncept crne kutije

ROS2 okvir koristi DDS (Data Distribution Service) kao osnovnu komunikacijsku sredinu, što omogućuje robusnu, skalabilnu i pouzdanu razmjenu podataka. Komunikacija se temelji na distribuiranoj arhitekturi koja omogućava razmjenu podataka između različitih čvorova (nodes). Ova komunikacija se odvija putem mehanizama komunikacije kao što su teme (topics), usluge (services) i akcije (actions), osiguravajući fleksibilnost i skalabilnost u raznovrsnim aplikacijama. Teme omogućuju asinkronu komunikaciju čvorova izdavača (eng. publisher) i preplatnika (eng. subscriber). Čvor putem izdavača (eng. publisher) može objavljivati poruke na određenoj temi, dok se drugi čvorovi mogu pretplatiti na tu temu putem preplatnika (eng. subscriber) kako bi primili poruke. Teme su korisne za kontinuirane tokove podataka. Nadalje, akcije omogućuju složenije interakcije koje uključuju višekratnu razmjenu podataka između čvorova tijekom dužeg vremenskog perioda. Akcije su korisne za operacije koje mogu trajati određeno vrijeme i zahtijevaju povratne informacije o napretku. Akcije se objedinjuju u action cli-

ent i action server mehanizmu komunikacije. Tok podataka između akcijskog klijenta (eng. action clienta) i akcijskog servera (eng. action server) u ROS 2 odvija se u nekoliko ključnih koraka: inicijalizacija, slanje cilja, povratne informacije, rezultat i eventualni prekid. Ovaj proces omogućuje složene interakcije između čvorova koje uključuju višekratnu razmjenu podataka, što je posebno korisno za operacije koje traju neko vrijeme i zahtijevaju kontinuirane povratne informacije. Čvorove pokreće launch datoteka koja može biti u XML ili Python formatu, omogućavajući korisnicima specifikaciju parametara, argumenata i konfiguracije potrebne za koordinaciju i pokretanje više komponenti u složenim ROS2 aplikacijama.

U ovome radu se koriste ROS2 mehanizmi komunikacije izdavača i pretplatnika za razmjenu podataka o geografskim podatcima mape prostora, upravljanje procesom postavljanja polazišne i odredišne točke putanje te u procesu pretvorbe podataka pogodnih za vizualizaciju unutar RViz2 alata. Mehanizam action client i action server s pripadajućim čvorovima akcijskoj klijenta i akcijskog servera koristi se u procesu planiranja putanje plovila.

Iznimno bitan koncept komunikacije u modelski informiranom globalnom planiranju putanje je i koncept crne kutije (eng. black-bx concept). Koncept crne kutije često se koristi u raznim poljima znanosti za opisivanje sustava, uređaja ili procesa čije unutarnje funkcioniranje nije vidljivo ili nije važno za razumijevanje njegovog ponašanja. Bitno je samo kako se sustav ponaša na temelju ulaznih podataka i kakve izlazne podatke isporučuje. U kontekstu ovoga rada, koncept crne kutije podrazumijeva da različiti paketi međusobno razmjenjuju informacije putem ROS2 mehanizama samo sa sadržajem geografskih koordinata. Koncept je usvojen zbog modularnosti koda, odnosno kako bi se svaki od navedenih paketa u budućim fazama projekta mogao zamijeniti s novijom naprednjom verzijom jer se informacije razmjenjuju u obliku geografskih koordinatama koje su standardne i prihvaćene u velikom spektru programskih rješenja. Naime, paketi koji nisu zamijenjeni trebali bi moći neometano izvršavati zadane funkcionalnosti te s novijom verzijom razmjenjivati informacije na jednak način kao i s prethodnom verzijom. Kako svaki paket interna stvara lokalni koordinati sustav na osnovi ulaznih podataka te pomoću njega vrši proračune te na kraju procesa koordinate lokalnog koordinatnog sustava prevara ponovno u geografske koordinate globalnog koordinatnog sustava ovako je

nešto moguće postići. Principi pretvorbe globalnog koordinatnog sustava geografskih koordinata i lokalnog koordinatnog sustava opisani su u poglavlju 2.2. S druge strane, ako bi paketi vršili komunikaciju isključivo s koordinatama lokalnog koordinatnog sustava, proces izrade planiranja putanje bi funkcionirao isključivo u svrhu prikaza koncepata ispitanih u radu, ali ne bi bio primjenjiv u stvarnom okruženju. Razlog tome je gubitak informacije o geografskim koordinatama polazišne i odredišne točke putanje te same generirane putanje plovila. Primjerice, ako bi navedeni paketi trebali isporučiti koordinate isplanirane putanje ili primiti geografske koordinate polazišne i odredišne točke putanje iz aplikacija koja nisu izrađena u okviru ovog projekta, takvo nešto ne bi bilo moguće. Razmjena informacija između aplikacija razvijenim od različitih proizvođača industrijski je standard te o tome izuzetno treba voditi brigu prilikom razvoja programske potpore. Naime, aplikacija za planiranje putanje plovila ne mora biti proizvedena od istog proizvođača kao sustav za vizualizaciju stanja različitih sastavnica plovila i sustava za upravljanje plovidbom. Međutim, navedeni sustavi moraju posjedovati mogućnost razmijene standardno prihvaćenih tipova podataka u što spadaju geografske koordinate. Također, prilagodba novijih verzija paketa u okviru ovog projekta bila bi višestruko zah-tjevnija jer bi se paket morao prilagoditi za rad s postavljenim lokalnim koordinatnim sustavom. Koncept crne kutije izuzetno je bitan kako bi programska potpora za planiranje putanje plovila mogla biti ispitana i funkcionirati u stvarnom okruženju kroz komunikaciju s vanjskim aplikacijama koje nisu izrađene u okviru ovog projekta te kako bi se svaki paket ovog projekta mogao zamijeniti naprednjom inaćicom bez prilagodbe ostalih paketa.

Zaključno, ovo poglavlje opisuje osnovne informacije i uloge ROS2 radnog prostora, paketa i mehanizama komunikacije u procesu modelski informiranog planiranja putanje poglavlja. Kroz naredna poglavlja navedeni koncepti će biti detaljno razrađeni kroz opise paketa, njihovih unutarnjih mehanizama komunikacije te mehanizama komunikacije s drugim paketima u ROS2 okviru.

2.2. Objasnjenje i pretvorbe između globalnog i lokalnog koordinatnog sustava

Koordinatni sustavi su sustavi koji prikazuju točke na pravcu, krivulji, plohi, ravnini ili prostoru pomoću brojeva, tj. koordinata, a predstavljaju temeljni alat u matematici, fizici, geografiji i mnogim znanstvenim disciplinama, omogućujući precizno definiranje položaja točaka u prostoru. Oni pružaju okvir unutar kojeg se mogu opisati položaji, kretanja i odnosi između objekata. Jedan od najpoznatijih primjera primjene koordinatnih sustava su mape. Mape su pojednostavljeni prikazi geografskog prostora, ističu odnose među objektima unutar prostora te prikazuju geografska obilježja poput reljefnih oblika, vodenih masa i umjetnih tvorevina. Znanost koja izučava izradu geografskih mapa naziva se kartografija. Kartografija koristi geografske koordinatne sustave, poput geografske širine i dužine za precizno određivanje lokacija na Zemljinoj površini. Takve mape su ključne i za planiranje putanje plovila koje se oslanja i bazira na koordinatne sustave za precizno određivanje pravca i smjera putanje plovila. Prije samog definiranja i objasnjenje modelski-informiranog planiranja putanje (poglavlje 2.5.) koje se zasniva na izrađenoj mapi prostora geografskih koordinata (poglavlje 2.3.) i postavljenoj polazišnoj i odredišnoj točci putanje (poglavlje 2.6.), potrebno je definirati pojmove globalnog i lokalnog koordinatnog sustava jer se navedeni procesi zasnivaju na pretvorbama između njih.

Globalni koordinatni sustav je definiran kao sustav geografskih koordinata (geografske širine i dužine) pomoću kojeg se prikazuju reljefi, vodene mase i obilježja na Zemljinoj površini. Globalni koordinatni sustav ovoga rada proizlazi iz OpenStreetMap mape prostora 2.3. u čijoj je osnovi *Pseudo-Mercator projekcija geografski koordinata (EPSG:3857)* koja predstavlja ključnu tehnologiju u suvremenoj kartografiji i geografskim informacijskim sustavima (GIS). Ova je projekcija poznata i kao *Web Mercator*, a ima široku uporabu pri digitalnom mapiranju i prikazivanju geografskih podataka na internetskim mapama prostora. Pseudo-Mercator projekcija je prilagodba klasične Mercatorove projekcije koja omogućuje prikazivanje Zemljine površine na ravnoj karti uz relativno jednostavne računske postupke prilagodbe geografskih koordinata, što ju čini izuzetno korisnom u digitalnim kartografskim aplikacijama što se koristilo i u izradi ovoga rada.

Lokalni koordinatni sustav proizlazi iz globalnog koordinatnog i temelji se na principu pretvorbe geografskih koordinata u metarsku skalu. Izrada lokalnog koordinatnog sustava započinje izdvajanjem dijela globalnog koordinatnog sustava u kojem će se vršiti obrada podataka. Obrada podataka variabilan je pojam te u različitim poglavljima poprima različita značenja. Obrada podataka u poglavljiju izrade mape prostora 2.3. poprima značenje obrade značajki slike za detekciju koordinata obale 2.3.3. te potom poravnjanja dobivenih koordinata izradom "kolaž" mape koordinata prostora 2.3.4. Kod modelski informiranog globalnog planiranja putanje 2.5. obrada podataka podrazumijeva izradu mape cijena prostora koja se temelji na izrađenoj mapi prostora te prilagodbu i primjenu algoritma za planiranje putanje plovila na osnovi mape cijena. Pretvorba globalnog u lokalni koordinatni sustav također se koristi i u prikazu mape unutar RViz2 vizualizacijskog alata 2.6.2. gdje obrada podataka naznačuje prilagodbu geografskih koordinata formatu prikladnom za vizualizaciju mape i karakteristika putanje plovila.

Izdvajanje dijela globalnog koordinatnog sustava predstavlja izdvajanje četiri rubne koordinate geografskog prostora koje naznačuju minimalne i maksimalne vrijednosti koordinata prostora. Sukladno navedenom, geografski prostor obrade podataka definira se kao geografski prostor između rubnih koordinata. Izdvojene se rubne koordinate koriste da bi se izračunala dužina i širina definiranog geografskog prostora u metrima. Funkcija pretvorbe geografskih koordinata u metre kao ulazne podatke prima četiri argumenta koji predstavljaju navedene rubne koordinate. Zadane koordinate u geografskim stupnjevima se potom pretvaraju u razliku koordinata u radijanima kako bi se mogla upotrijebiti **Haversine formula** za izračunavanje velike kružne udaljenosti između dvije tačke na sferi, uzimajući u obzir zakrivljenost Zemlje (jednadžbe 2.1 i 2.2). Udaljenost se izračunava kao umnožak parametra C dobivenog iz Haversine formule te polujmjera Zemlje R što se na posljetku množi s 1000 kako bi se vrijednost pretvorila u metarsku skalu (jednadžba 2.3). Haversine formula glasi:

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (2.1)$$

$$c = 2 \cdot \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) \quad (2.2)$$

$$d = R \cdot c \cdot 1000 \quad (2.3)$$

Gdje su:

ϕ_1, λ_1 = Geografska širina i dužina prve koordinate

ϕ_2, λ_2 = Geografska širina i dužina druge koordinate

$$\Delta\phi = \phi_2 - \phi_1$$

$$\Delta\lambda = \lambda_2 - \lambda_1$$

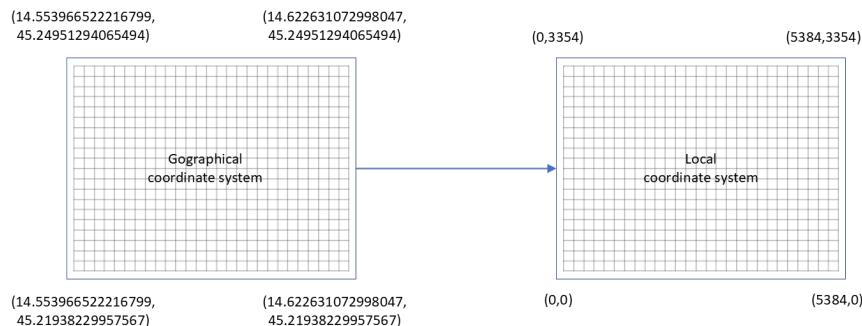
R = Radijus zemlje (mean radijus = 6,371 km)

d = Udaljenost dvije točke na površini sfere

Haversine formula je izuzetno korisna za računanje udaljenosti između dviju točaka na Zemlji kada su koordinate izražene u geografskoj širini i dužini. Međutim, Pseudo-Mercator projekcija koristi ravninske koordinate, što zahtijeva prilagodbu proračuna udaljenosti. Iako se Pseudo-Mercator koristi za prikazivanje geografskih mapa na ravnim plohamama, izračunavanje udaljenosti u ovom sustavu može biti izazovno zbog izobličenja i neravnina na većim geografskim prostorima. Uzveši u obzir navedeno, potrebno je nglasiti da su u ovome radu korišteni geografski prostori na kojima je utjecaj zakrivljenosti Zemljine površine na projekciju koordinata zanemariv. Shodno navedenom, korištena je Haversine formula bez prilagodbe koordinata Pseudo-Mercator projekciji. U većim geografskim prostorima u kojem zakrivljenost Zemljine površine nije zanemarivo, koordinate dobivene Pseudo-Mercator projekcijom geografskog prostora moraju biti prilagođene kako ne bi došlo do pogrešne interpretacije geografskih koordinata.

Veličina lokalnog koordinatnog sustava definirana je duljinom i širinom rubnih geografskih koordinata u metarskoj skali dobivenih pomoću Haversine formule. Primjerice, rubne geografske koordinate $14.553966522216799, 45.21938229957567$ 14.622631072998047 i 45.24951294065494 čine dio prostora globalnog koordinatnog sustava i osnova su stvaranja lokalnog koordinatnog sustava pretvorjom iz globalnog. Karakteristike su lokalnog koordinatnog sustava da se najmanja geografska koordinata ($14.553966522216799, 45.21938229957567$) translatira u koordinatu $(0,0)$, a najveća ge-

ografska koordinata (14.622631072998047 i 45.24951294065494) u koordinatu (55384,3364) koja odgovara geografskog dužini i širini u metarskoj skali dobivenih pomoću prethodno opisane Haversine formule (jednadžbe 2.1 do 2.3). Ovakvom pretvorbom globalnog u lokalni koordinati sustav stvara se prostor koordinata (eng. grid) u kojem je moguće odrediti značajke u lokalnom koordinatnom sustavu čije su koordinate poznate i u globalnom koordinatnom sustavu. Jedan takav primjer je određivanje koordinata obale iz skalirane fotografije u kojoj jedan piksel fotografije odgovara jednom metru kvadratnom u prirodi što će biti detaljnije opisano u poglavlju 2.3.3. Postupak stvaranja lokalnog koordinatnog sustava iz globalnog koordinatnog sustava korištenjem Haversine formule prikazan je slikom 2.1.



Slika 2.1. Primjer pretvorbe globalnog koordinatnog sustava u lokalni koordinatni sustav

Značajan parametar cijelokupnog procesa izrade mape i planiranja putanje plovila je **parametar uzorkovanja** (eng. *grid_size*) jer se u njegovoj ovisnosti vrši stvaranje mape prostora uzorkovanjem koordinata obale što je opisano u poglavljima 2.3.3. i 2.3.4. na čijem principu se zasniva testiranje algoritama za planiranje putanje plovila 2.5.2. i planiranje konačne putanje plovila temeljenog na mapi cijena 2.5.3. Parametar uzorkovanje definira korak u metrima (pikselima), odnosno koliko se svakih metara u prirodi uzima uzorak geografskih koordinata. U ovome radu korišten je parametar uzorkovanja od 10 metara koji odgovara preporuci da bi parametar uzorkovanja trebao odgovarati duljini plovila za koje se planira putanja kako bi vrijeme reakcije i izvedivost putanje bili optimalni. Za potrebe rada prepostavljeno je da će sustav biti testiran na plovilu srednje veličine, ali je parametar uzorkovanja varijabilan i izrađeni računalni kod može računati s različitim cjelobrojnim parametrima uzorkovanja u ovisnosti o veličini plovila. Definiranje parametara uzorkovanja osnovni je preduvjet optimizacije pretvorbe lokalnog i globalnog koordinatnog sustava.

Kako bi pretvorba koordinata uopće i bila moguća, potrebno je definirati i opisati funkcije pretvorbe. Jednadžbe pretvorbe geografskih koordinata u koordinate lokalnog koordinatnog sustava zasnivaju se na principu pretvorbe i poravnjanja koordinata uz odabrani parametra uzorkovanja ($grid_size:=g$). Jednadžbe pretvorbe geografskih koordinata u koordinate lokalnog koordinatnog sustava prikazane su u jednadžbama 2.4 i 2.5 Iako ove koordinate prikazuju točnu pretvorbu koordinata, potrebno ih je prilagoditi parametru uzorkovanja. Shodno tome, prostor koordinata u lokalnom koordinatnom sustavu određen je izborom parametra uzorkovanja te je ostatak dijeljenja tih koordinata s parametrom uzorkovanja jednak nuli (kongurencija) tj. $(x,y) = (x,y) * g$, gdje je (x,y) proizvoljna koordinata definiranog lokalnog koordinatnog sustava. Uz odabrani cjelobrojni parametar uzorkovanja g , prostor koordinata odgovara brojevima djeljivim s g . Zbog prethodno navedenog, koordinate koje nisu djeljive s parametrom uzorkovanja se zaokružuju (eng. round) na prvi manji ili veći broj djeljiv bez ostatka sa zadanim brojem (jednadžbe 2.6 i 2.7). Primjerice, koordinata (12,28) se zaokružuje na koordinatu (10,30), a koordinata (38, 42) na (40,40) po principu zaokruživanja. Uzorkovanjem koordinata jednim dijelom se gubi vjernost prikaza koordinata jer različite geografske koordinate poprimaju jednake vrijednosti u lokalnom koordinatnom sustavu. S druge strane, smanjuje se računska složenost procesa pretrage koordinata za g puta te uz odabrani parametar uzorkovanja koji je jednak ili manji duljini plovila, gubitak određenih vrijednosti koordinata ne utječe značajno na proces planiranja putanje.

Pretvorba koordinata iz lokalnog koordinatnog sustava u geografsku širinu i dužinu vrši se jednadžbama 2.8 i 2.9 koje su zapravo izvedenice iz jednadžbi 2.4 i 2.5 te vrijedi obrat. Zbog poravnjanja koordinata u lokalnom koordinatnom sustavu gubi se također dio informacija (koordinata) u globalnom sustavu, ali kao što je prethodno spomenuto uz pravilan odabir parametra uzorkovanja gubitak koordinata ne utječe na daljnje procese. Nadalje, gubitak određenih koordinata prilikom planiranja putanje nadomješta se postupkom zaglađivanja putanje (eng. curve fitting) što je opisano u poglavlju 2.5.4.

$$x_{\text{local}} = \left\lfloor \frac{(\lambda_{\text{longitude}} - \lambda_{\min}) \cdot s_x}{\lambda_{\max} - \lambda_{\min}} \right\rfloor \quad (2.4)$$

$$y_{\text{local}} = \left\lfloor \frac{(\phi_{\text{latitude}} - \phi_{\min}) \cdot s_y}{\phi_{\max} - \phi_{\min}} \right\rfloor \quad (2.5)$$

$$x_{\text{local}} = \left\lfloor \frac{x_{\text{local}}}{g} \right\rfloor \cdot g \quad (2.6)$$

$$y_{\text{local}} = \left\lfloor \frac{y_{\text{local}}}{g} \right\rfloor \cdot g \quad (2.7)$$

$$\phi_{\text{latitude}} = \phi_{\min} + y_{\text{local}} \cdot \frac{\phi_{\max} - \phi_{\min}}{s_y} \quad (2.8)$$

$$\lambda_{\text{longitude}} = \lambda_{\min} + x_{\text{local}} \cdot \frac{\lambda_{\max} - \lambda_{\min}}{s_x} \quad (2.9)$$

gdje su:

$(\lambda_{\text{longitude}}, \phi_{\text{latitude}})$: koordinata globalnog koordinatnog sustava

$(x_{\text{local}}, y_{\text{local}})$: koordinata lokalnog koordinatnog sustava

λ_{\min} : minimalna geografska širina

λ_{\max} : maksimalna geografska širina

ϕ_{\min} : minimalna geografska dužina

ϕ_{\max} : maksimalna geografska dužina

s_x : širina geografskog prostora u metrima(pikselima)

s_y : dužina geografskog prostora u metrima(pikselima)

g : parametar uzorkovanja (grid size)

Programska izvedba preračuna globalnog i lokalnog koordinatnog sustava u različitim inačicama temelji se na stvaranju objekta klase pretvorbe koordinatnih sustava. Objektno orijentirana klasa pretvorbe u konstruktoru inicijalizira objekt s nekoliko atributa temeljenih na ulaznim parametrima. Ulazni parametri su rubne koordinate pre-

dane direktno ili se one određuju kao minimalne i maksimalne vrijednosti geografske širine i dužine iz predanih lista koordinata te parametar uzorkovanja (eng. grid_size). Na osnovu inicijaliziranog objekta stvara se mapa prostora (eng. grid). Ulazni parametri variraju ovisno o primjeni te posljedično i pojedine funkcije zadane klase. Osim rubnih koordinata i parametra uzorkovanja, zajedničke su funkcije svim programima i funkcije pretvorbe globalnih koordinata u lokalne i obrnuto u ovisnosti o parametru uzorkovanja. Primjene i prilagodbe osnovnog koncepta pretvorbe koordinatnih sustava će biti opisane u narednim poglavljima. Međutim, potrebno je voditi računa o tome da sve pretvorbe slijede ista osnovna pravila i logiku opisanu u ovome poglavlju kako bi se postigla modularnost, jednostavnost i praktičnost koda.

Kao što je već prethodno navedeno, pretvorbe između globalnog i lokalnog koordinatni sustav s manjim preinakama koriste se gotovo u svakom koraku izrade modelski informiranog globalnog planiranja putanje plovila zbog modularnosti koda. Pojam modularnosti koda podrazumijeva da se u budućim koracima projekta svaka sastavnica može zamijeniti novom i naprednijom inačicom bez da zamjena utječe na ostale sastavnice. Primjerice, u narednim koracima projekta se sustav za izradu mape prostora obradom podataka s OpenStreetMap mape može zamijeniti određenom verzijom digitalne pomorske karte koja sadržava više potrebnih informacija za planiranje putanje plovila. Uz određene preinake postojećeg algoritma i pravilno povezivanje pomorske mape s ROS2 okvirom, postojeći sustav za planiranje putanje plovila trebao bi u potpunosti funkcioniрати. Kako bi se postigla modularnost koda, potrebno je usvojiti koncept crne kutije (eng. black-box concept) kako bi sustavi međusobno razmjenjivali isključivo informacije geografskih koordinata što je detaljnije objašnjeno u poglavlju 2.1.3. Koncept crne kutije nije bio neophodan za prikaz teorijskih koncepata ovoga rada, ali bez njega bi se izgubile informacije o geografskim koordinatama te nastali kod ne bi bio primjenjiv za praktičnu upotrebu u stvarnome okruženju.

2.3. Izrada pomorske mape geografskih koordinata

Prvi korak za uspješnu izradu sustava za globalno planiranje putanje plovila je izrada mape geografskog prostora s naznačenim hidrografskim obilježjima. Mape geografskog prostora koje se koriste za planiranje putanje i navigaciju plovila u industriji i znanstvenom području pomorske robotike jesu pomorske mape. Pomorska je mapa po definiciji geografska mapa određenog plovidbenog područja (dijela zemaljske površine), konstruirana u jednoj od usvojenih kartografskih projekcija, a mora sadržavati sve elemente potrebne za orientaciju i sigurnu navigaciju plovila. Za sigurnu navigaciju plovila, što podrazumijeva i planiranje putanje plovila, značajna su hidrografska obilježja morskog dijela mape kao što su morfologija morskog dna, koordinate obale, udaljenost od iste te naznačena područja opasnosti plovidbe. U trenutku izrade diplomskog rada, autoru nisu bile dostupne digitalne pomorske mape kao niti drugi oblici digitalnih mapa prostora kao što su OpenStreetMap i Google maps interaktivne mape dostupne putem API ključeva. Zbog prethodno navedenog, izrađeni su programi za obradu slike i dostupnih HTML podataka s javno dostupne OpenStreetMap mape u cilju izrade mape prostora s hidrografskim obilježjima koordinata obale i udaljenosti od iste. Također, zbog nedostatka informacija u istu nisu uključena hidrografska obilježja morfologije dna i područja opasnosti plovidbe, ali u narednim će koracima projekta njihovo uključivanje biti značajna stavka.

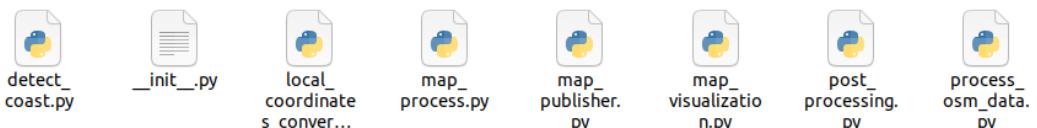
Kroz ovo poglavlje je opisan postupak izrade mape prostora obradom preuzetih podataka s OpenStreetMap internetske stranice upotrebom pretvorbe koordinatnih sustava opisanog u prethodnom poglavlju 2.2. te povezivanje dobivene mape s ROS2 okvirom.

2.3.1. Struktura i pokretanje ROS2 paketa `map_maker`

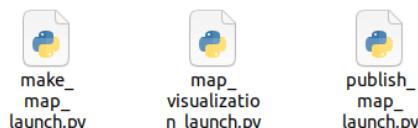
Za razumijevanje tijeka i uloge pojedinih programa u procesu izrade mape prostora, potrebno je objasniti strukturu ROS2 paketa **map_maker**. ROS2 paket map_maker dijeli strukturu s ostalim ROS2 paketima te pripada path_planning_ws ROS2 radnom prostoru (eng. workspace) što je opisano u poglavlju 2.1. Paket ima zadaću stvaranja mape geografskog prostora obradom podataka i povezivanje dobivene mape s ROS2 okvirom putem ROS2 izdavača (eng. ROS2 publisher). Struktura paketa te pripadajućih programskih mapa map_maker i launch prikazani su u nastavku na slikama 2.2., 2.3. i 2.4.



Slika 2.2. Prikaz strukture ROS2 paketa map_maker



Slika 2.3. Prikaz računalne mape map_maker



Slika 2.4. Prikaz računalne mape launch

Prva zadaća paketa je izrada mape prostora geografskih koordinata obradom fotografije i HTML zapisa s OpenStreetMap mape spremljenih u računalnu mapu *input_data*. Detaljni proces izrade i struktura mape su opisani u poglavljima od 2.3.2. do 2.3.4. Glavni program koji upravlja procesom stvaranja mape prostora je *map_process.py*. Program *map_process.py* je ujedno i ROS2 čvor, pokreće se putem ROS2 okvira te prima parametre iz *make_map_launch.py* launch datoteke. Navedeni parametri su *save_file_name* naziv pod kojim se spremi izrađena mapa, parametar uzorkovanja *grid_size* i imena programske mape *locations* iz *input_data*. Programske mape koje čine *input_data* sadržavaju podatke preuzete s OpenStreetMap mape čija je obrada osnovni korak ovoga procesa. Nadalje, podaci spremljeni u programske mape trebali bi biti podatci susjednih geografskih prostora koji se jednim dijelom poklapaju kako bi se postigla homogenost prostora izrađene krajnje mape. Svaki od navedenih programa u nastavku ujedno je i klasa po načelu objektno orijentiranog programiranja te se za svaki program stvara i pripadajući objekt u *map_process.py* programu.

Zadaća ROS2 čvora *map_process.py* je koordiniranje toka programa spremanjem i obradom međurezultata te spremanje procesirane mape prostora u *map_data* programske mape. Nakon inicijalizacije parametara iz launch datoteke stvara se programska petlja čiji broj iteracija ovisi o broju inicijaliziranih programske mape *locations*. Prvi program koji u programskoj petlji poziva *map_process.py* je *process_osm_data.py*. Navedeni program služi za obradu podataka iz programske mape *input_data* što podrazu-

mijeva stvaranje objekta lokalnog koordinatnog sustava 2.2. te skaliranje fotografije na jedan piksel po kvadratnome metru geografskog prostora (poglavlje 2.3.2.). Nadalje, poziva se program *detect_coast.py* koji iz skalirane fotografije detektira koordinate obale i zona udaljenosti od nje te korištenjem prethodno izrađenog objekta pretvorbe koordinatnih sustava pretvara koordinate iz lokalnog u globalni koordinatni sustav. Rezultati iteracija programske petlje spremaju se u listu geografskih koordinata obale i liste zona udaljenosti od nje.

Završetkom programske petlje poziva se klasa *post_processing.py* koja u konstruktoru inicijalizira izrađene liste geografskih koordinata te stvara lokalni koordinatni sustav za poravnanje dobivenih koordinata 2.3.4. Poravnate se koordinate lokalnog koordinatnog sustava potom ponovno pretvaraju u koordinate globalnog koordinatnog sustava i u obliku lista vraćaju glavnemu programu. Naposljetku, glavni program *map_process.py* iz obrađenih podataka stvara rječnik (eng. dictionary) s predefiniranom strukturom te isti spremi u binarnu datoteku naziva *processed_map_save_file_name* gdje je *save_file_name* definirano u launch datoteci. Binarna se mapa spremi u programsku mapu *map_data* čime završava proces izrade mape geografskog prostora. Za vizualizaciju dobivene mape koristi se *map_visualisation_launch.py* launch datoteka.

Druga je zadaća paketa povezivanje dobivenih mapa s ROS2 okvirom putem ROS2 izdavača. ROS2 čvor *map_publisher.py* obrađuje rječnike binarnih datoteka sa sadržajem geografskih koordinata prostora te ih pretvara u ROS2 poruku *CoastMsg.msg* te putem ROS2 izdavača naziva 'gps_coordinates_coast' objavljuje u ROS2 okruženju. Detaljniji je opis postupka dostupan u poglavlju 2.3.5. Logički prikaz toka programa izrade mape prostora i objavljivana ROS2 poruke putem ROS2 izdavača s pripadajućom legendom je prikazan na slici2.5.

Popis ROS2 naredbi za pokretanje map_maker paketa

Za izradu mape prostora je potrebno definirati parametre *save_file_name*, *locations* i *grid_size* u *make_map_launch.py* datoteci, izgraditi ROS2 paket *path_planning_ws* te u Linux terminalu pokrenuti naredbu :

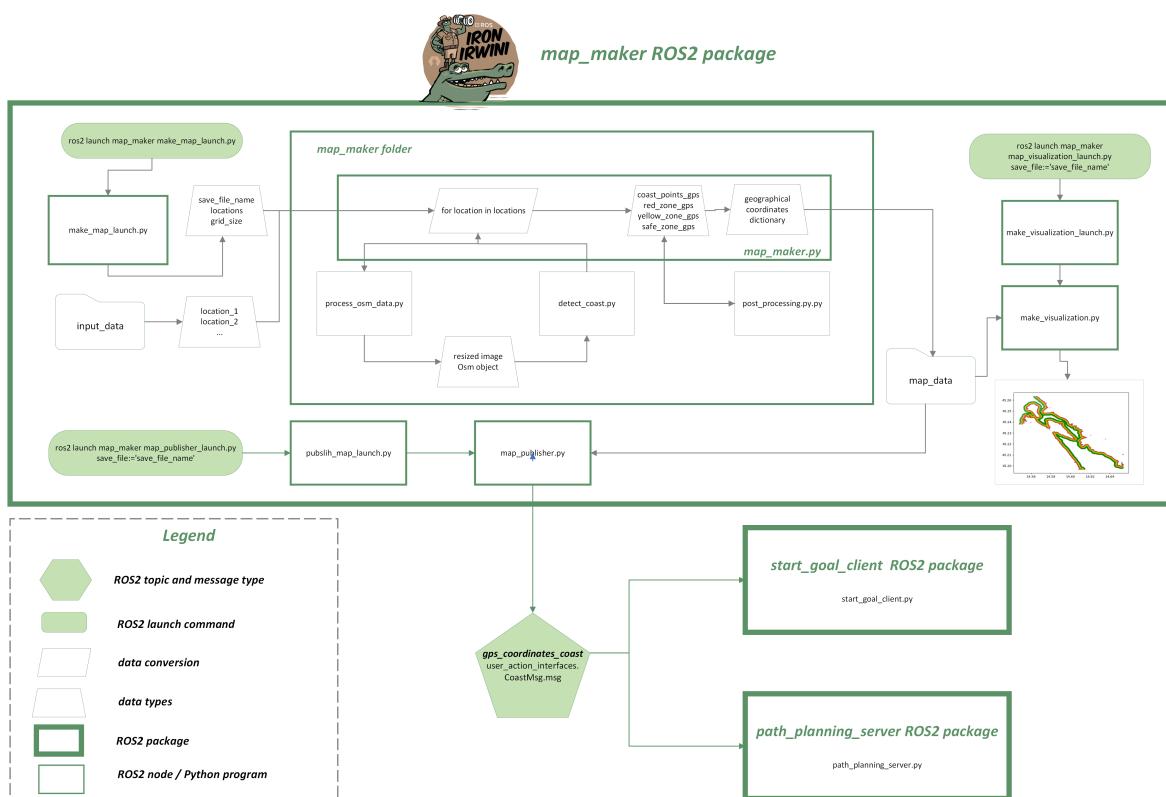
```
ros2 launch map_maker make_map_launch.py
```

Za vizualizaciju dobivenih mapi prostora se koristi sljedeća naredba u Linux terminalu gdje save_file_name parametar naznačuje ime željene mape za vizualizaciju iz definirane launch datoteke:

```
ros2 launch map_maker make_visualization.launch.py
save_file:='save_file_name'
```

Kako bi dobivenu mapu objavili putem ROS2 izdavača 'gps_coordinates_coast' potrebno je pokrenuti sljedeću naredbu u Linux terminalu gdje save_file_name parametar naznačuje ime željene mape za objavljivanje iz launch datoteke:

```
ros2 launch map_maker publish_map.launch.py
save_file:='save_file_name'
```



Slika 2.5. Arhitektura komunikacije ROS2 paketa `map_maker`

2.3.2. Preuzimanje i obrada podataka s OpenStreetMap internetske stranice

OpenStreetMap (OSM) je javno dostupna geografska baza podataka koja koristi Pseudo-Mercator (Web Mectaor) kartografsku projekciju što je detaljnije opisano u prethodnom poglavlju 2.2. Sama web stranica OpenStreetMap je online karta, tražilica i uređivač geopodataka. OpenStreetMap nudi mogućnost pretrage područja u različitoj rezoluciji, tj. nadmorskoj visini (eng. zoom level). Dostupna je pretraga područja u 20 metara, 30 metara, 50 metara, 100 metara i 300 metara nadmorske visine pri kojima je očuvana detaljnost prikaza topografskih obilježja reljefa. Prikazi područja ispod 300 metara nadmorske visine detaljno prikazuju obilježja reljefa, ali je prikazani geografski prostor zadane fotografije relativno malen. Posljedično, broj potrebnih fotografija za prikaz većeg područja se višestruko povećava čime raste složenost obrade podataka računalnog procesa. S druge strane, nedostatak prikaza područja na nadmorskim visinama većim od 300 metara, uz manje detaljan prikaz reljefa je i nemogućnost skaliranja fotografije po principu jedan piksel fotografije po metru kvadratnom prostora jer premašuje maksimalni dozvoljeni broj piksela za prikaz fotografije, što je bitna stavka u dalnjem postupku izrade mape. Sukladno prethodno navedenom, za obradu podataka odabran je prikaz u nadmorskoj visini od 300 metara zbog očuvanja topografskih značajki reljefa te mogućnosti skaliranja fotografije na jedan piksel fotografije po kvadratnome metru.

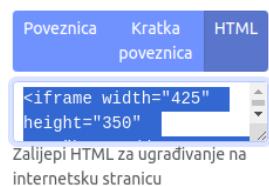
Postupak preuzimanja podataka je sljedeći: unutar mape *input_data* koja je dio ROS2 programskog paketa *map_maker* izrađuju se mape geografskog prostora (eng. folder) koje nose naziv određenog geografskog područja, u dalnjem tekstu naznačeno kao *lokacija*. Unutar svake mape *lokacija* nalaze se dvije datoteke koje je potrebno preuzeti s OpenStreetMap internetske stranice s postavkom od 300 metara nadmorske visine (eng. 300 m zoom level) u odjeljku *Podijeli*. Prva stavka je fotografija u *.png* formatu koju je potrebno preuzeti (slika 2.6. te potom spremiti u računalnu mapu *lokacija* pod nazivom *lokacija.png=*). Druga je stavka kreirati tekstualna datoteka *osm_info.txt* (u svim računalnim mapama unutar računalne mape *input_data* nosi isti naziv), kopirati HTML zapis iz odjeljka *Podijeli* (slika 2.8.) te ga zalijepiti u izrađenu datoteku *osm_info.txt*. Primjer preuzete fotografije *jadranovo* i HTML zapisa iz mape *lokacija:=jadranovo* nalaze se na slikama 2.7. i 2.9.



Slika 2.6. Preuzimanje fotografije s OpenStreetMap



Slika 2.7. Preuzeta fotografija *jadranovo.png* rezolucije 2237×1226 piksela



Slika 2.8. Preuzimanje HTML zapisa s OpenStreetMap

```
<iframe width="425" height="350" src="https://www.openstreetmap.org/export/embed.html?bbox=14.570703506469728%2C45.20852893922364%2C14.642372131347658%2C45.235854890331254&layer=napnlk" style="border: 1px solid black"></iframe><br><small><a href="https://www.openstreetmap.org/#map=15/45.2222/14.6065">Prikaži veću kartu</a></small>
```

Slika 2.9. Izgled preuzetog HTML zapisa iz *osm_info.txt* datoteke

Kao što je opisano u prethodnom poglavlju 2.3.1., ROS2 čvor *process-osm-data.py* prvo inicijalizira parametre iz launch datoteke *save_file_name* naziv pod kojim se sprema izrađena mapa, parametar uzorkovanja *grid_size* i imena programskih mapa *locations* iz *input_data* te stvara programsku petlju obrade podataka. Parametri se potom unutar programske petlje prosljeđuju klasi *process_osm_data.py* koja je zadužena za skaliranje fotografije i obradu zapisa *osm_info.txt* datoteke. Postupak obrade preuzetih podataka opisanih u prethodnom dijelu poglavlja unutar *process_osm_data.py* započinje s obradom HTML zapisa iz *osm_info.txt* u kojem su naznačene geografske širine i dužine rubnih koordinata fotografije. Dio Html zapisa koji sadrži podatke rubnih koordinata započinje s naznakom "bbox=" te završava s naznakom "&" , dok su koordinate odvojene naznakama "%2C". Primjer jednog takvog zapisa glasi :

```
bbox=14.570703506469728%2C45.20852893922364
%2C14.642372131347658%2C45.235854890331254&
```

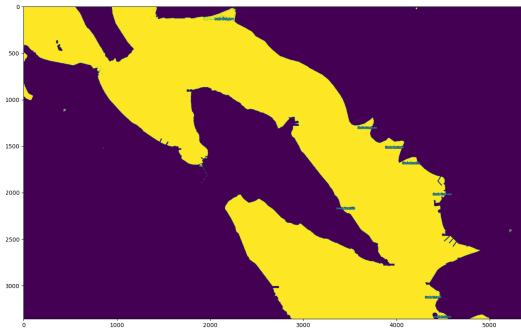
Gdje su 14.570703506469728, 45.20852893922364, 14.642372131347658 i 45.235854890331254 izdvojene rubne koordinate geografskog prostora.

Prema prethodno opisanim načelima pretvorbe globalnog koordinatnog sustava u lokalni koordinatni sustav iz poglavlja 2.2., rubne se koordinate koriste za stvaranje lokalnog koordinatnog sustava. Nadalje, stvara se računalni objekt pretvorbe u kojem je moguće pristupiti funkcijama pretvorbe koordinatnih sustava. Bitna stavka koja nije detaljno opisana u poglavlju 2.2. je skaliranje fotografije po načelu jedan piksel po kvadratnome metru. Navedeno načelo skaliranje fotografije kazuje točan odnos između pozicije piksela na fotografiji i odgovarajuće geografske koordinate u stvarnosti, što u kasnijim koracima omogućuje jednostavno pretraživanje značajki slike u lokalnom koordinatnom sustavu te pretvorbu značajki u globalni koordinatni sustav. Obrada preuzetih podataka završava sa skaliranjem fotografije koja se uz objekt pretvorbe prosljeđuje natrag glavnom programu *process-osm-data.py* te se pristupa obradi značajki skalirane fotografije što je opisano u narednom poglavlju.

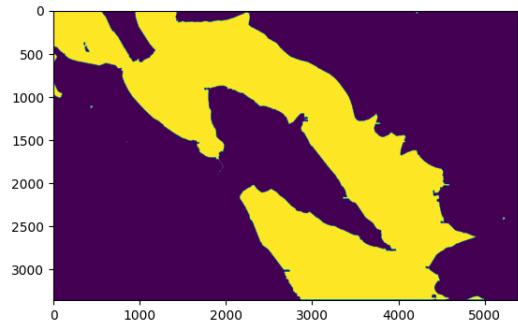
2.3.3. Pronalaženje koordinata obale obradom fotografije i postavljanje zona udaljenosti

Pronalaženje koordinata obale obradom fotografije i postavljanje zona udaljenosti od obale je proces u čijoj je osnovi obrada značajki skalirane fotografije iz prethodnog poglavlja te poznavanje odnosa lokalnog koordinatnog sustava fotografije i globalnog koordinatnog sustava kroz pristup izrađenom objektu pretvorbe koordinatnih sustava. Za obradu fotografije te potom pretvaranje njezinih značajki u globalni koordinati sustav zadužena je klasa *detect_coast.py* kojoj se u početku prosljeđuju skalirana fotografija, objekt pretvorbe i parametar uzorkovanja.

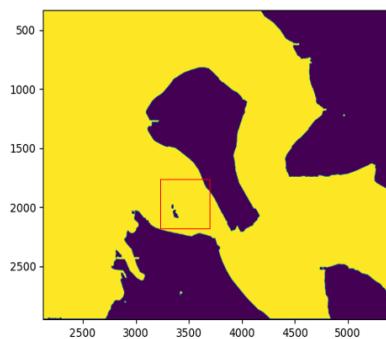
Proces započinje pretvaranjem skalirane fotografije iz RGB (Red, Green, Blue) prostora boja u HSV (Hue, Saturation, Value) prostor boja pomoću CV2 Python modula koji se koristi za obradu slike. HSV prostor boja se smatra prihvatljivijim za ljudsko razumevanje i manipulaciju bojama u usporedbi s tradicionalnim prikazima. Metodom pokušaja i pogrešaka je ustanovljeno da je raspon plave boje koja odgovara moru na zadanoj fotografiji u rasponu od HSV vrijednosti [80, 60, 60] do HSV vrijednosti [120, 255, 255]. Uvezši u obzir granice plave boje prikaza mora na fotografiji, stvara se binarna maska u kojoj su svi pikseli unutar definiranog raspona postavljeni na vrijednost 255 (bijelo), a svi ostali na 0 (crno). Ovakvim se prikazom postiže da su pikseli prikaza mora prikazani jednom vrijednošću, a pikseli značajki kopna ujednačeni i prikazani drugom vrijednošću.



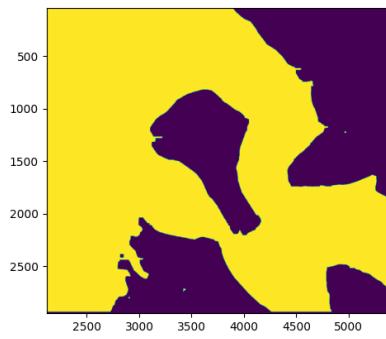
Slika 2.10. Prikaz binarne maske bez postupka binarnog zatvaranja



Slika 2.11. Prikaz binarne maske s postupkom binarnog zatvaranja



Slika 2.12. Prikaz binarne maske s veličinom matrice 12x12



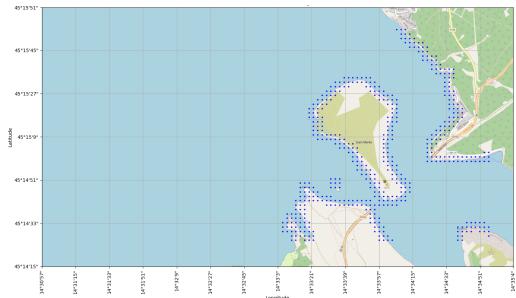
Slika 2.13. Prikaz binarne maske s veličinom matrice 30x30

Zbog prikaza određenih topoloških naziva na području mora, fotografiju je potrebno dodatno obraditi postupkom binarnog zatvaranja kako bi se nazivi uklonili. Po definiciji binarno zatvaranje je morfološka operacija koja kombinira dilataciju i eroziju, što pomaže u uklanjanju malih crnih rupa unutar bijelih područja maske i spajanju bijelih dijelova koji su blizu jedan drugom. Prikaz dobivene binarne maske sa i bez primjenom binarnog zatvaranja prikazani su na slikama 2.10. i 2.11. Kod binarnog zatvaranja treba pripaziti na veličinu matrice zatvaranja jer ako je matrica veća od dimenzija 15x15 dolazi do gubitka geografskih obilježja poput manjih otoka i hridi. Primjer je prikazan na slikama 2.12. i 2.13. gdje se zbog prevelike dimenzije matrice izgubila informacija o otočiću Selahovac u blizini Krčkog mosta (na slici naznačen crvenim kvadratom) što može biti veliki sigurnosni problem prilikom planiranja putanje broda.

Proces određivanja koordinata obale i zona udaljenosti od iste temelji se na očitavanju vrijednosti piksela binarne maske s primjenom binarnog zatvaranja. Pretraga se vrši na način da se pretražuju rubne vrijednosti područja određenog vrijednosti piksela 0 (ljubičasto područje na slikama) ili 255 (žuto područje na slikama), tj. traže se koordinate



Slika 2.14. Prikaz uzorkovanih točaka obale uz parametar uzorkovanja 10



Slika 2.15. Prikaz uzorkovanih točaka obale uz parametar uzorkovanja 50

prijelaza iz vrijednosti 0 u 255 što se definira kao koordinata obale. Pretraživanje koordinata obale vrši se u ovisnosti o parametru uzorkovanja `grid_size` tako da se pretražuju samo one koordinate (i njih susjedne koordinate) čiji je ostatak dijeljenja s parametrom uzorkovanja jednak nuli kako bi se smanjila računska složenost procesa. Prepostavka pretrage u ovisnosti o parametru uzorkovanja funkcioniра onda kada je vrijednost parametra uzorkovanja relativno mala. Međutim, za veće parametre uzorkovanja algoritam je potrebno prilagoditi kako ne bi došlo do gubitka informacija o koordinatama obale. Primjer gubitka informacija o koordinatama obale prikazan je na slikama 2.14. i 2.15. gdje su uzorkovane točke obale prikazane plavim krugovima. Navedeno ukazuje na važnost činjenice da parametar uzorkovanja obale ne bude veći od 20 jer u suprotnom dolazi do netočnog određivanja i gubitka informacija o koordinatama obale. Isto tako, prilikom planiranja putanje većih polovila (duljina 20 metara i više) zbog prethodno navedenog problema gubitka informacija ne može biti pravilo da parametar uzorkovanja bude jednak duljini broda. Stoga je bolje prilagoditi korak pretrage algoritma planiranja putanje plovila nego li uzorkovati obalu većim parametrom uzorkovanja te posljedično izgubiti točne informacije o koordinatama obale što također predstavlja sigurnosni problem.

Pravila plovidbe koja propisuje Ministarstvo mora, prometa i infrastrukture Republike Hrvatske kazuju da se plovila ne smiju približavati obali na manje od 50 metara ukoliko ne uplovjavaju ili isplovjavaju iz područja luke. Nadalje, na udaljenosti 150 m od obale svi plovni objekti dužni su ploviti s posebnom pažnjom, brzinom ne većom od 5 čvorova, a na udaljenosti od 150 m do 300 m od obale svi plovni objekti dužni su ploviti s posebnom pažnjom brzinom ne većom od 8 čvorova, ukoliko nije drugačije propisano. Na osnovi propisanih pravila definirane su crvena zona (0 do 50 metara od obale), žuta zona (50 do 150 metara od obale) i zelena zona (150 do 300 metara od obale) što je osnova

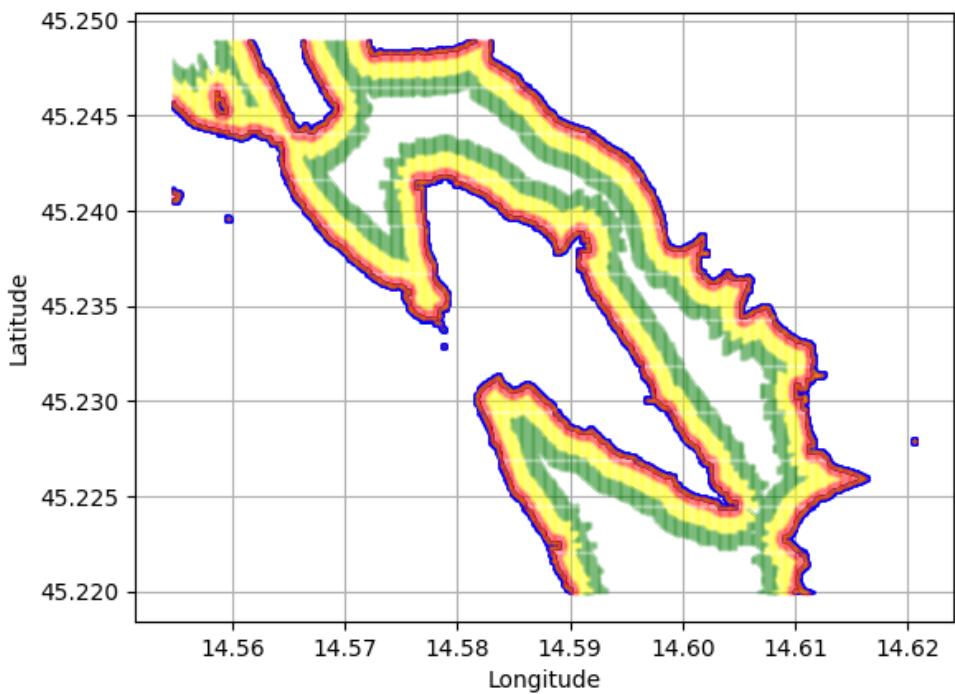
za izradu mape cijena za planiranje putanje broda i upotreba će biti detaljnije opisana u poglavlju 2.5.3. Naposljeku, definirana je i sigurna zona planiranja putanje (300 do 350 m) koja je nužna u postupku zaglađivanja putanje te će također biti detaljnije opisana u poglavlju 2.5.4. U narednim slikama nije prikazana sigurna zona jer je sastavnica internog planiranja sigurne putanje,a ne zakonskog propisa.

Postupak određivanja koordinata zona se temelji na obradi prethodno određenih geografskih koordinata obale. Za svaku se koordinatu obale izračunavaju koordinate na određenim euklidskim udaljenostima od nje u definiranom broju smjerova (kružna pretraga). Za manje detaljan prikaz zona udaljenosti dovoljno je odabrat 12 ili 24 smjerova pretrage, ali za optimalan prikaz područja bez nehomogenih dijelova prostora u zonama na većim udaljenostima potrebno je izabrati minimalno 36 smjerova. Prikaz zona udaljenosti od obale uz 12 smjerova pretrage koje sadrži nehomogen prikaz zelene zone te 36 smjerova pretrage koje rješava problem nehomogenosti zona prikazani su na slikama 2.16. i 2.17. Smjerovi se definiraju kao (n je broj željenih smjerova):

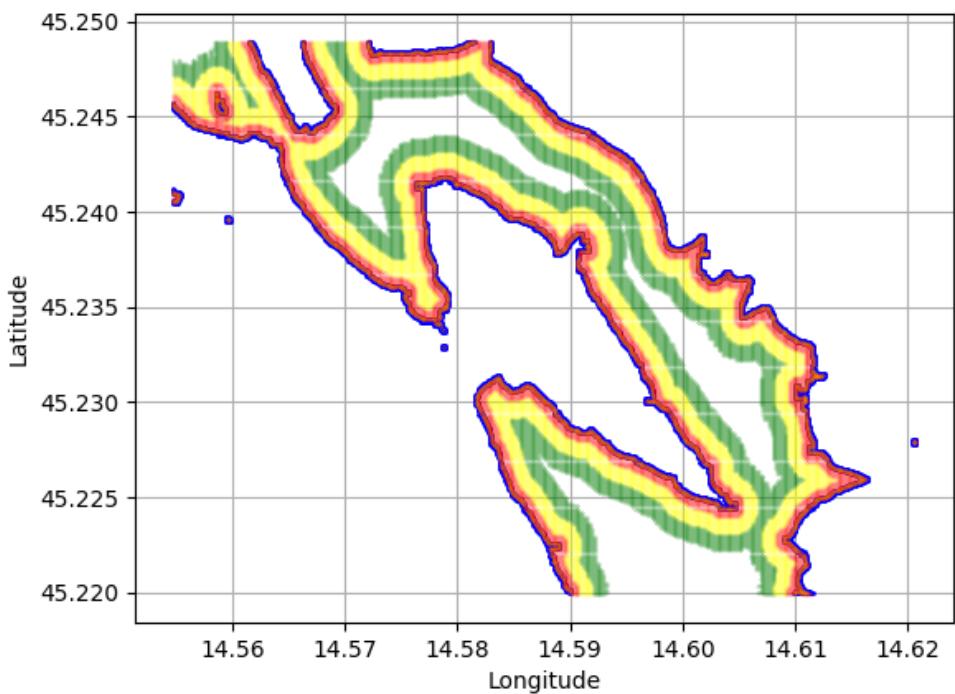
$$\text{directions} = \left[\left(\cos\left(\frac{2\pi}{n}i\right), \sin\left(\frac{2\pi}{n}i\right) \right) \mid i = 0, 1, \dots, n - 1 \right] \quad (2.10)$$

Za svaki se smjer potom gleda euklidska udaljenost od zadane koordinate pa se s obzirom na dobivenu vrijednost koordinate pridaje crvenoj, žutoj, zelenoj ili sigurnoj zoni. Potrebno je naglasiti da se u procesu vrši dodatna provjera valjanosti koordinate. Naime, za svaku se dobivenu koordinatu provjerava njegina vrijednost na binarnoj maski. Ukoliko je vrijednost piksela koordinate na binarnoj maski 255, koordinata se proglašava kao valjana jer predstavlja koordinatu mora, a u suprotnom se odbacuje.

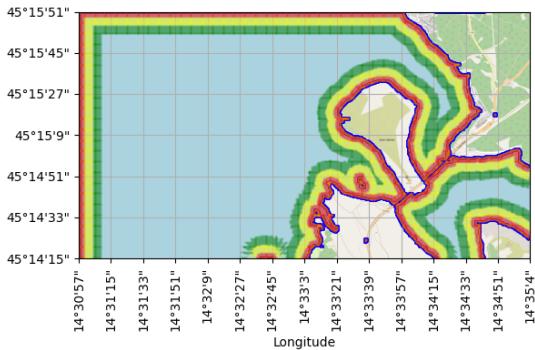
Postupak određivanja koordinata obradom fotografije donosi određene probleme na koje je potrebno pripaziti te ih ukloniti. Prvi problem je što određene fotografije preuzete s OpenStreetMap stranice sadržavaju crne piksele (vrijednost piksela 0) na rubovima fotografije te se skaliranjem iste njihov broj povećava. Crni pikseli fotografije imaju negativan utjecaj jer ih program prepoznaje kao točku obale, dok u stvarnosti to nije slučaj. Primjer pogrešno prepoznate obale zbog crnih piksela rubova fotografije nalazi se na slici 2.18. Prvi bi pristup u rješavanju ovog problema bio smanjivanje početne fotografije za iznos crnih piksela.



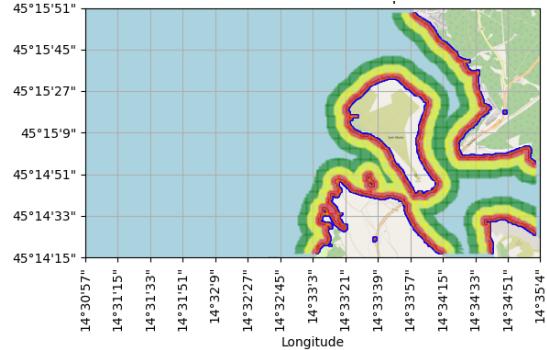
Slika 2.16. Prikaz određenih zona udaljenosti od obale uz 12 smjerova pretrage



Slika 2.17. Prikaz određenih zona udaljenosti od obale uz 36 smjerova pretrage



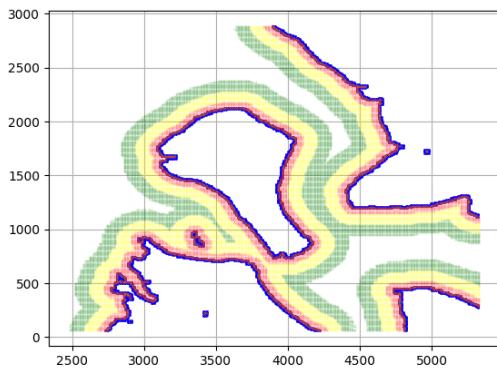
Slika 2.18. Prikaz problema obrade slike rubnih crnih piksela i prikaz mosta



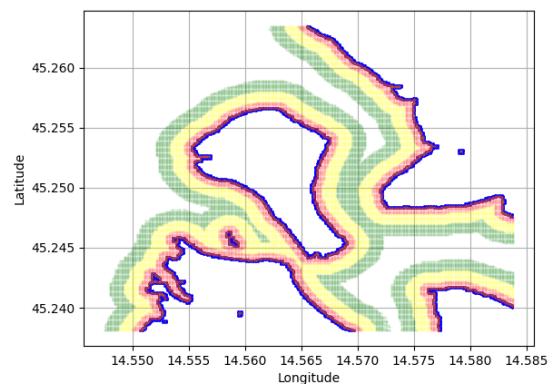
Slika 2.19. Prikaz rješenja problema obrade slike

Međutim, zbog postupka pretvaranja koordinatnih sustava ovakvo nešto nije moguće izvesti bez stvaranja pomaka između prikazane koordinate u lokalnom i globalnom koordinatnom sustavu koji kao posljedicu ima pogrešno iščitavanje koordinata. Zbog prethodno navedenog, ovaj problem je riješen na način da se koordinate koje su na udaljenosti manjoj od 50 piksela od rubova fotografije odbacuju, tj. ne uzima ih se u obzir prilikom određivanja koordinata obale i koordinata zona. Problem koji se također pojavljuje je prikaz mostova. Naime, mostovi su prikazani dvodimenzionalno i algoritam ih prepoznaje kao točke obale što se također može vidjeti na slici 2.18. gdje je Krčki most pogrešno interpretiran kao obala. Ovu vrstu problema rješava se obradom fotografije u nekom od programa za obradu fotografije poput Adobe Photoshopa gdje se jednostavno most postavlja u boju jednaku nijansi boje mora. Postavka algoritma da se odbacuju pikseli na rubovima fotografije te grafički postupak obrade fotografije uspješno su riješili prethodno opisane probleme što je prikazano na slici 2.19.

Posljednji je dio procesa obrade fotografije pretvaranje dobivenih lokalnih koordinata (piksela) obale i zona udaljenosti od obale u geografske koordinate postupkom opisanim u poglavljju 2.2. koristeći zadani objekt (klasu) programa *process_osm_data.py*. Primjer prikaza rezultantnih koordinata u lokalnom 2.20. te globalnom 2.21. koordinatnom sustavu. Naposljetku se dobivene liste geografskih koordinata obale i zona udaljenosti vraćaju putem poziva funkcije glavnog programu *map_process.py* te se nastavlja programska petlja za obradu podataka i značajki fotografije iz ostalih definiranih programske mape u launch file datoteci (iz *input_data* programske mape).



Slika 2.20. Prikaz resultantnih koordinata u lokalnom koordinatnom sustavu



Slika 2.21. Prikaz resultantnih koordinata u globalnom koordinatnom sustavu

2.3.4. Izrada kolaž mape geografskih koordinata postupkom poravnjanja koordinata i spremanje rezultata

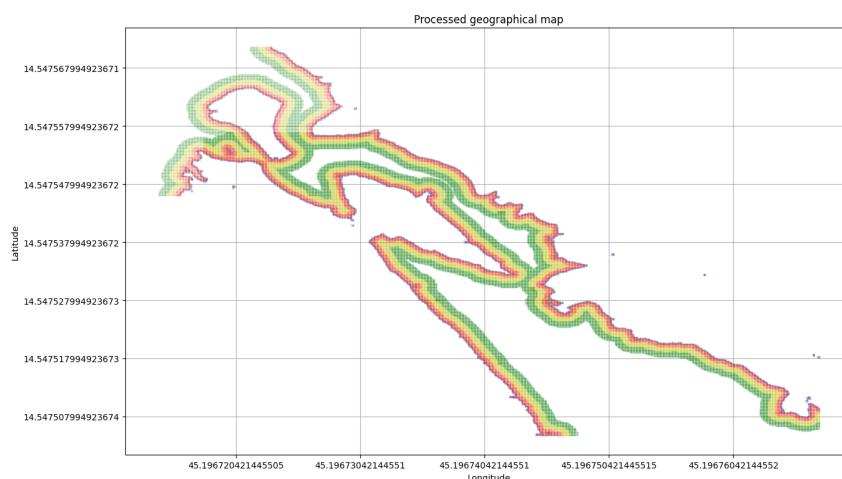
Posljednji je korak izrade mape geografskog prostora poravnanje koordinata dobivenih obradom značajki fotografije u prethodnom poglavlju, odnosno izrada kolaž mape geografskih koordinata. Pojam kolaž se koristi kako bi se naznačilo da je dobivena mapa zapravo nastala spajanjem obrađenih značajki različitim fotografijama geografskog prostora u jedan cjeloviti prikaz prostora. ROS2 čvor *map_process.py* u programskoj petlji obrađuje podatke preuzete s OpenStreetMap stranice te u svakoj iteraciji dobivene geografske koordinate obale i zona udaljenosti od obale sprema u zajedničke kolaž liste geografskih koordinata. Navedene liste su liste geografskih koordinata obale, crvene, žute, zelene i sigurne zone čije su koordinate prikazane u obliku geografskih širina i dužina (globalni koordinatni sustav). Problem navedenih lista koordinata je što koordinate nisu poravnjene, odnosno njihov prikaz je raspoređen u prostoru u ovisnosti o rubnim koordinatama slike i parametru uzorkovanja. Takav nehomogen prikaz prostora u kojima nisu ujednačeni zapisi geografskih širina i dužina potrebno je prilagoditi kako bi se dobio homogen prikaz koordinata pogodan za primjenu algoritama planiranja putanja plovila.

Poravnanje koordinata dobivenih listi vrši se u klasi *post_processing.py* koja u konstruktoru inicijalizira predane liste geografskih koordinata i parametar uzorkovanja što je preduvjet stvaranja lokalnog koordinatnog sustava po principu opisanom u poglavlju 2.2. Nadalje, liste obale, crvene, žute, zelene i sigurne zone inicijaliziranih koordinata se potom spajaju u veliku listu koordinata. Iz navedene liste izdvajaju se maksimalne

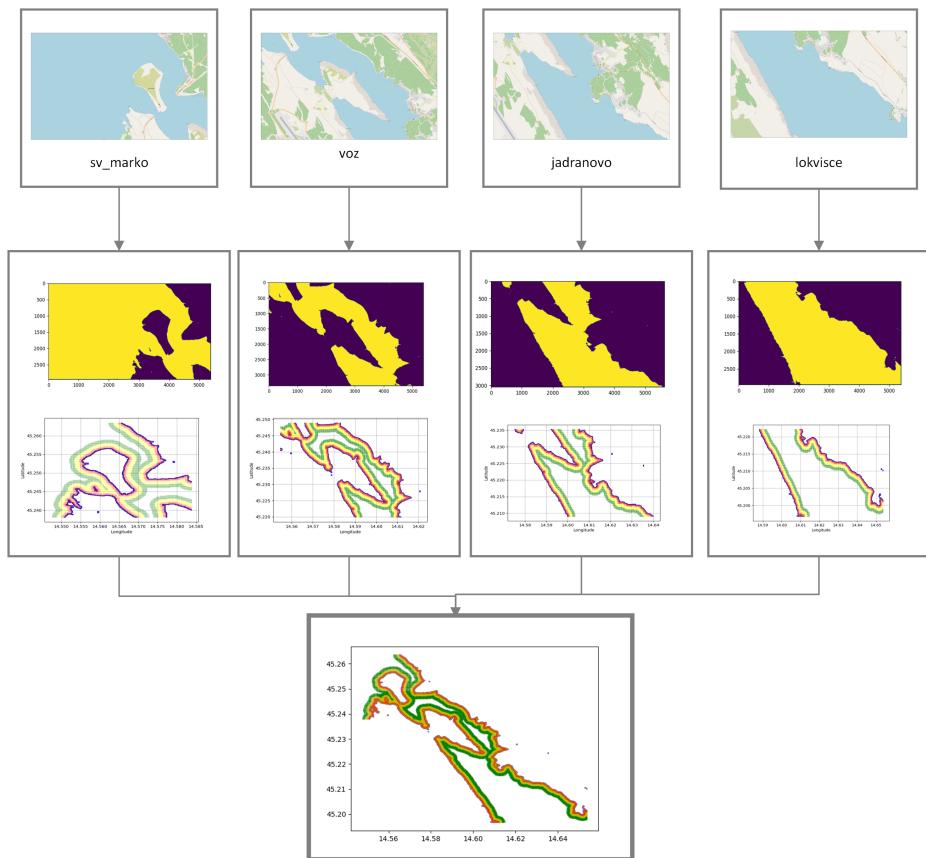
i minimalne globalne vrijednosti geografskih koordinata (rubne koordinate prostora) te se pomoću Haversine formule stvara lokalni koordinatni sustav. Uz pridajući lokalni koordinatni sustav također se inicijaliziraju funkcije pretvorbe koordinatnih sustava.

Nakon stvaranja lokalnog koordinatnog sustava obrađuju se liste geografskih koordinata te se s obzirom na parametar uzorkovanja pomoću funkcija pretvorbe pretvaraju u lokalni koordinatni sustav (jednadžbe 2.4 do 2.7). Ovom se pretvorbom postiže poravnanje koordinata jer koordinate koje nisu poravnate u globalnom koordinatnom sustavu se pretvorbom u lokalni koordinatni sustav postavljaju u koordinate koje dijeljenjem s parametrom uzorkovanja imaju ostatak nula. Odnosno, susjedne koordinate geografskih područja, čiji se prikaz na fotografijama poklapa, se translatiraju u jednake koordinate te se dobiva homogen prikaz prostora. Shodno tome, ovim se postupkom uklanja višak koordinata nastao preklapanjem prikaza geografskih područja te geografske koordinate prostora poprimaju uniforman homogen prikaz.

Naposljeku se dobivene liste poravnatih koordinata u lokalnom koordinatom sustavu pomoću funkcija pretvorbe pretvaraju u liste geografskih koordinate globalnog koordinatnog sustava te se prosljeđuju nazad glavnome programu pozivom funkcije čime je proces poravnjanja koordinata i stvaranja kolaž mape završen. Prikaz kolaž mape geografskog prostora nastale poravnanjem koordinata prikazan je na slici 2.22., a pogledostavljeni grafički prikaz cjelokupnog postupka stvaranja mape prostora obradom značajki fotografija prikazan je na slici 2.23.



Slika 2.22. Grafički prikaz kolaž mape geografskog prostora



Slika 2.23. Grafički prikaz postupka obrade fotografije i stvaranja kolaž mape prostora

Posljednja je stavka procesa izrade mape geografskog prostora spremanje iste u binarnu datoteku. Binarna datoteka nosi naziv *processed_map_save_file_name* gdje je *save_file_name* definirano u launch datoteci te se spremaju u programsku mapu *map_data*. Dobivene geografske koordinate prostora spremaju se u rječnik (eng. dictionary) s definiranom strukturu prikazanom u tablici 2.1. u obliku tablice u nastavku te se rječnik potom spremaju u binarnu datoteku čime završava cijelokupni proces izrade mape prostora geografskih koordinata.

Ključ (key)	Potključ (key)	Vrijednost (value)
header	-	naslov
time_stamp	-	vrijeme izvedbe programa
processed_areas	-	obrađene programske mape
data	grid_size	parametar uzorkovanja
data	coast_points_gps	koordinate obale
data	red_zone_gps	koordinate crvene zone
data	yellow_zone_gps	koordinate žute zone
data	green_zone_gps	koordinate zelene zone
data	safe_zone_gps	koordinate sigurne zone

Tablica 2.1. Tablica strukture rječnika za prikaz koordinata mape geografskog prostora

2.3.5. Objavljivanje geografskih koordinata mape unutar ROS2 okvira

U prvotnim je poglavlјima opisan postupak izrade mape geografskog prostora. Izrađene geografske mape se spremaju u binarnu datoteku *processed_map_save_file_name* gdje je *save_file_name* definirano u launch datoteci u obliku rječnika čija je struktura prikaza u tablici 2.1. Navedene binarne datoteke rječnika koje sadržavaju informacije o geografskim koordinatama prostora se nalaze u programskoj mapi *map_data*.

Povezivanje izrađenih mapa s ROS2 okvira vrši se putem čvora *map_publisher.py* kojemu se preko launch datoteke ili Linux terminala predaje ime željene mape za objavljanje. Nadalje, inicijalizira se izdavač (eng. publisher) pomoću kojeg će biti dostupni podaci mape geografskog prostora putem teme (eng. topic) *gps_coordinates_coast*. Čvor potom preuzima rječnik geografskih podataka iz binarne datoteke spremljene u programskoj mapi *map_data*. Preuzeti s rječnika obrađuje i pretvara u poruku *CoastMsg.msg* iz *user_action_interfaces* paketa. Paket *user_action_interfaces* unutar okvira ima ulogu definiranja poruka i akcija. Struktura ROS2 poruke *CoastMsg.msg* prikazana je u tablici 2.2. Kako bi pretvorba geografskih koordinata iz formatu rječnika bila uspješno prilagođena formatu poruke, potrebno je razdvojiti koordinate geografske širine i dužine jer ROS2 okvir ne podržava više dimenzijska polja. Postupak završava objavom *CoastMsg.msg* putem *map_publisher.py* te je sadržaj poruke dostupan putem teme '*gps_coordinates_coast*'.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
std_msgs/Float32MultiArray	coast_points_x
std_msgs/Float32MultiArray	red_points_x
std_msgs/Float32MultiArray	yellow_points_x
std_msgs/Float32MultiArray	green_points_x
std_msgs/Float32MultiArray	safe_points_x
std_msgs/Float32MultiArray	coast_points_y
std_msgs/Float32MultiArray	red_points_y
std_msgs/Float32MultiArray	yellow_points_y
std_msgs/Float32MultiArray	green_points_y
std_msgs/Float32MultiArray	safe_points_y
float32	grid_size

Tablica 2.2. Struktura ROS2 poruke *CoastMsg.msg*

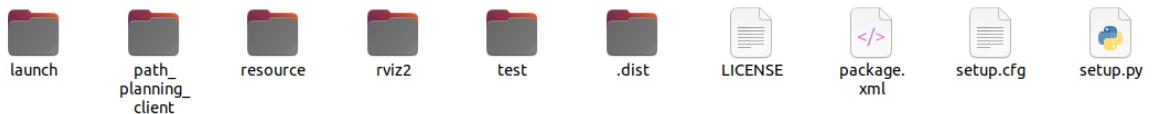
2.4. Postavljanje polazišne i odredišne točke planiranja putanje plovila i vizualizacija podataka

Planiranje se putanje plovila odnosi na proces određivanja optimalne rute pomoću različitih tehnika i algoritama od točke polazišta do točke odredišta. U samoj definiciji planiranja putanje plovila ističe se nekoliko pojmove. Prvi pojmovi su tehnike i algoritmi pod što spadaju algoritmi za planiranje putanje te tehnike interpolacije dobivene putanje što je opisano u poglavlju 2.5. Međutim, prije nego li se pristupi procesu planiranja putanje plovila, kao što sama definicija kazuje, potrebno je postaviti polazišnu i odredišnu točku putanje. U ovome će poglavlju upravo biti opisane tehnike postavljanja polazišne i odredišne točke planiranja putanje plovila. Između ostalog, bit će opisane tehnike vizualizacije navedenih točaka i isplanirane putanje na dobivenoj mapi geografskog prostora iz prethodnoga poglavlja 2.3. unutar RViz2 vizualizacijskog alata.

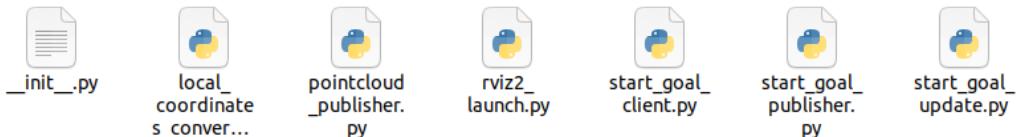
2.4.1. Struktura i pokretanje ROS2 paketa path_planning_client

ROS2 paket *path_planning_client* zadužen je za vizualizaciju podataka unutar RViz2 vizualizacijskog alata, upravljanje procesom postavljanja polazišne i odredišne točke te objavljivanje istih unutar ROS2 okvira putem action client i action server mehanizma komunikacije za planiranje putanje plovila. Struktura paketa te pripadajućih programskih mapa launch ipath_planning_client prikazani su u nastavku na slikama 2.45., 2.46. i 2.47.

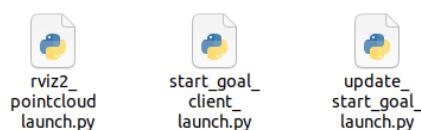
Vizualizacija geografskih koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, polazišne i krajnje točke putanje, interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje vrši se pomoću čvora *poincloud_publisher.py*. Detaljniji opis i tehničku izvedbu postupka vizualizacije moguće je pronaći u poglavljima 2.6.2., 2.6.3. i 2.6.4. Čvor inicijalizira preplatnike te pomoću preplatnika *start_goal_publisher* dohvata globalne koordinate geografskog prostora objavljene na temi *gps_coordinates_coast* putem izdavača *publish_map_launch.py* iz *map_maker* paketa. Shodno navedenom, preduvjet dohvata podataka je pokretanje čvora *publish_map_launch.py*. Pomoću dohvaćenih geografskih koordinata globalnog koordinatnog sustava tvori se lokalni koordinatni sustav u klasi *local_coordinates_converter.py* po principu opisanom u poglavlju 2.2. te se potom globalne koordinate pretvaraju u ko-



Slika 2.24. Prikaz strukture ROS2 paketa path_planning_client



Slika 2.25. Prikaz računalne mape path_planning_client



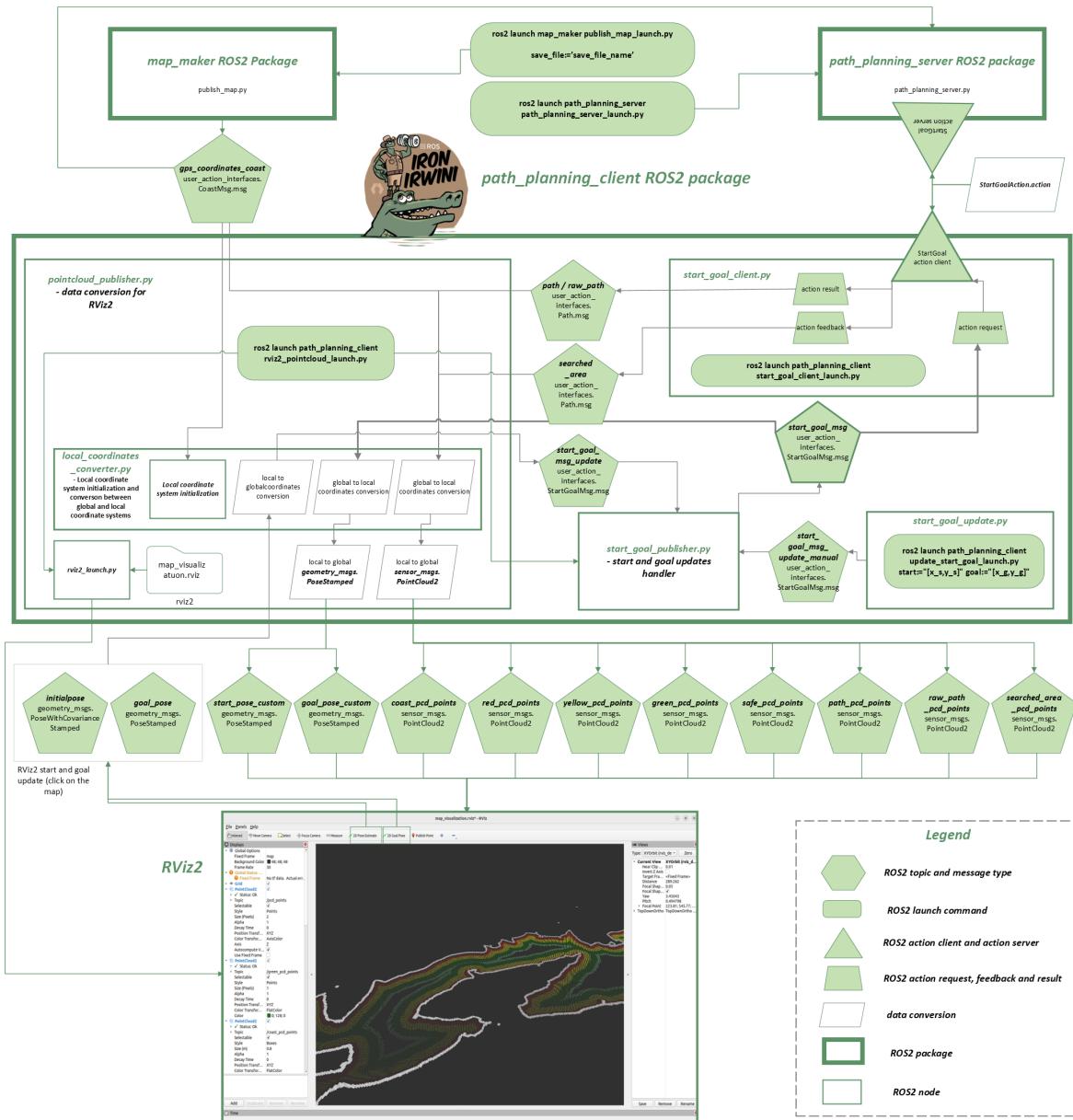
Slika 2.26. Prikaz računalne mape launch

ordinate lokalnog koordinatnog sustava. Međutim, koordinate lokalnog koordinatnog sustava nisu namijenjene prikazu u RViz2 alatu (jer su spremljene u obliku lista) već su među korak pretvorbe koordinata globalnog koordinatnog sustava u *sensor_msgs.PointCloud2* ili *geometry_msgs.PoseStamped* poruku. Kako je ranije navedeno, lokalne koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje se pretvaraju u poruku *sensor_msgs.PointCloud2*, a polazišne i krajnje točke putanje u poruku *geometry_msgs.PoseStamped*. Nadalje, pretvorene koordinate se pomoću izdavača objavljuju na pripadajućim temama namijenjenim za vizualizaciju u Rviz2 alatu.

Čvor *pointcloud_publisher.py* pokreće launch datoteka *rviz2_pointcloud_launch.py*. Osim čvora pretvorbe podataka za vizualizaciju, launch datoteka poziva čvor *rviz_launch.py* koji pokreće RViz2 vizualizacijski alat sa spremlijenim postavkama vizualizacije te čvor *start_goal_publisher.py* koji je zadužen za upravljanje procesom postavljanja polazišne i odredišne točke putanje. Nakon inicijalizacije pretplatnika, čvor *start_goal_publisher.py* se pretplaćuje na teme *start_goal_msg_update* i *start_goal_msg_update_manual* te s obzirom na aktualnost objavljenih podataka objavljuje polazišnu i odredišnu točku putanje na temu *start_goal_msg*. Pojam aktualnosti se odnosi na vrijednost točaka te vremenski

okvir njihovog postavljanja što je objašnjeno u poglavlju 2.6.3. Temu *start_goal_msg_update* objavljuje čvor *rviz2_pointcloud_launch.py* te sadrži podatke i vremenski okvir postavljenih točaka putanje unutar RViz2 alata. Nadalje, temu *start_goal_msg_update_manual* objavljuje čvor *start_goal_update.py* koji pokreće launch datoteka *update_start_goal.launch.py*. Pomoću launch datoteke se navedenom čvoru prosljeđuju podaci o točkama iz Linux terminala te zato tema ima sufiks *manual* kako bi se naznačilo da se podatke korisnik "ručno" objavljuje pokretanjem datoteke. Postoje mogućnosti ažuriranja samo polazišne ili odredišne točke putanje ili obje istovremeno, a čvor je moguće pokrenuti više puta sve dok korisnik nije zadovoljan postavljenim točkama. Zaključno, čvor *start_goal_publisher.py* prima ažuriranja polazišne i odredišne točke iz RViz2 alata i Linux terminala te se aktualna ažuriranja točaka postavljaju kao globalna polazišna i odredišna točka. Drugačije rečeno, moguće je primjerice postaviti polazišnu točku unutar RViz2 alata i odredišnu točku pomoću *start_goal_update.py* čvora te će navedene točke biti postavljene kao globalna polazišna i odredišna točka, iako su postavljene iz različitih izvora.

Postavljanje polazišne i odredišne točke se vrši u svrhu globalnog planiranja putanje plovila kojim presjeda action client i action server ROS2 mehanizam komunikacije. U trenutku nakon što je korisnik zadovoljan postavljenim globalnim točkama *start_goal_msg*, pokreće se launch datoteka *start_goal_client.launch.py* koji pokreće *start_goal_client.py* klijentski čvor. Navedeni čvor inicijalizira polazišnu i odredišnu točku putanje dostupne na temi *start_goal_msg* u vremenskom okviru pokretanja datoteke te akciju za planiranje putanje *start_goal_action*. U polje zahtjev inicijalizirane akcije *StartGoalAction.action* postavljaju se potom točke putanje te se pokreće asinkroni mehanizmi upravljanja komunikacijom. Mehanizam potom čeka pokretanje akcijskog servera *path_planning_server.py* iz *path_planning_server* paketa. Uspostavom komunikacije, server će kao ulazne podatke inicijalizirati točke iz zahtjeva te u povratnoj petlji vraćati područje pretrage algoritma planiranja putanje i napisljetu vratiti rezultat dobivene putanje u interpoliranom i neinterpoliranom obliku. Detaljniji opis komunikacije akcijskog klijenta i servera su opisani u poglavljima 2.6.3. i 2.6.4. Grafički prikaz arhitekture komunikacije akcijskog čvorova, mehanizma akcijskog klijenta i server te obrade i pretvorbe podataka uz navedene naredbe za pokretanje paketa *path_planning_client* prikazan je na slici 2.48.



Slika 2.27. Arhitektura komunikacije ROS2 paketa `path_planning_client`

Popis ROS2 naredbi za pokretanje ROS2 paketa `path_planning_client`

Za pokretanje RViz2 vizualizacijskog alata koristi se `rviz2_pointcloud_launch.py` launch datoteka. Launch datoteka poziva pripremljenu RViz2 datoteku `map_visualization.rviz` iz `rviz2` programske mape sa spremlijenim postavkama za vizualizaciju zadanih tema te ranije navedeni čvor za pretvorbu podataka `pointcloud_publisher.py`. Launch datoteka se pokreće naredbom:

```
ros2 launch path_planning_client rviz2_pointcloud_launch.py
```

Uz navedenu naredbu potrebno je pokrenuti i objavljivanje mape geografskog prostora iz **map_maker** paketa što je osnova stvaranja lokalnog koordinatnog sustava:

```
ros2 launch map_maker publish_map_launch.py  
save_file:= 'save_file_name'
```

Za pokretanje ažuriranja polazišne i odredišne točke putanje pomoću Linux terminala pokreće se naredba gdje x_s, y_s, x_g i y_g predstavljaju geografske koordinate polazišne i odredišne točke putanje :

```
ros2 launch path_planning_client update_start_goal_launch.py  
start:=[x_s,y_s]"
```

```
ros2 launch path_planning_client update_start_goal_launch.py  
goal:=[x_g,y_g]"
```

```
ros2 launch path_planning_client update_start_goal_launch.py  
start: "[x_s,y_s]" goal:=" [x_g,y_g]"
```

Ažuriranje točaka moguće je i direktno putem RViz2 alata klikom na polja alatne trake *2D Pose Estimation* i *2D goal pose* te postavljanjem oznaka na željene lokacije prikazane mape geografskog prostora.

Naposljetku, ukoliko je korisnik zadovoljan postavljenom polazišnom i odredišnom točkom, pokreće se sljedeća naredba, čija je svrha uspostava komunikacije putem akcijskog klijenta s akcijskim servom zaduženim za planiranje putanje plovila:

```
ros2 launch path_planning_client start_goal_client_launch.py
```

2.4.2. Vizualizacija podataka u RViz2 vizualizacijskom alatu

Shodno ranije navedenoj definiciji uloge *path_planning_client* paketa iz prethodnog poglavlja 2.45., prva je zadaća paketa vizualizacija željene mape geografskih koordinata, polazišne i odredišne točke (koordinate) te isplanirane putanje plovila. Vizualizaciju pokreće launch datoteka *rviz2_poincloud_launch.py* koja poziva rviz datoteku *map_visualization.rviz* (iz *rviz2* programske mape) sa spremlijenim postavkama za vizualizaciju geografskih koordinata obale, zona udaljenosti od obale, polazišne i odredišne točke putanje, područja pretrage algoritma planiranja putanje te isplanirane putanje u interpoliranom i neinterpoliranom obliku. Nadalje, poziva se čvor *pointcloud_publisher.py* koji vrši prilagodbu i objavljivanje podataka pogodnih za vizualizaciju u RViz2 alatu te čvor *start_goal_publisher* koji je zadužen za upravljanje postavljanja polazišne i odredišne točke putanje, o čemu će biti više rečeno u poglavlju 2.6.3. Kako bi se podaci uistinu mogli vizualizirati, potrebno ih je pretvoriti u oblik pogodan za vizualizaciju za što je zadužen čvor *pointcloud_publisher.py*. Pokretanjem launch datoteke, u konstruktoru čvora se inicijaliziraju pretplatnici na teme čije će poruke biti dohvaćene te potom pretvorene u druge vrste poruka pogodnih za vizualizaciju. Naziv preplaćenih tema s pripadajućim vrstama poruke i opisom podataka prikazani su u tablici 2.9.

Naziv teme	Vrsta poruke	Opis podataka
'gps_coordinates_coast'	CoastMsg	Koordinate obale i zona udaljenosti
'start_goal_msg'	StartGoalMsg	Geografske koordinate polazišne i odredišne točke
'path'	PathMsg	Geografske koordinate interpolirne putanje
'raw_path'	PathMsg	Geografske koordinate neinterpoliranog algoritma
'searched_area'	PathMsg	Podaci o pretraženom području algoritma za planiranje putanje
'initialpose'	PoseWith Covariance Stamped	Postavljena polazišna točka u RViz2
'goal_pose'	PoseStamped	Postavljena odredišna točka u RViz2

Tablica 2.3. Tablica preplaćenih poruka sa nazivima tema, vrstama poruka i opisima

Iz stupca *Vrste poruka* tablice 2.9. je vidljivo da osim poruke CoastMsg.msg čija je struktura definirana u poglavlju 2.3.5. postoje vrste poruka koje je tek potrebno definirati, poput CoastMsg.msg, poruke *StartGoalMsg.msg* i *PathMsg.msg* također su dijelom *user_action_interfaces* ROS2 paketa. Nadalje, poruke *PoseWithCovarianceStamped* i *PoseStamped* su standardne poruke ROS2 okvira te se njihov sadržaj može pronaći na ROS2 službenim internetskim stranicama.

Poruka *StartGoalMsg.msg* služi za prikaz ažuriranih geografskih koordinata polazišne i odredišne točke putanje te globalne polazišne i odredišne točke putanje plovila. Sadržaj globalne polazišne i odredišne točke putanje akcijski klijent predaje akcijskom serveru u svrhu planiranja putanje plovila što je opisano u poglavlju 2.6.3. Sadržaj poruke prikazan je u tablici 2.10.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32[]	start
float32[]	goal

Tablica 2.4. Struktura ROS poruke *StartGoalMsg.msg*

Poruka *PathMsg.msg* se koristi za prikazivanje dobivene putanje plovila u interpoliranom i neinterpoliranom obliku koje u polju rezultat vraća akcijski server akcijskom klijentu. Osim za objavu rezultata planirane putanje, poruka se koristi i za prikazivanje područja pretrage algoritma planiranje putanje koje akcijski server također kao povratnu informaciju vraća akcijskom klijentu. Sadržaj poruke prikazan je u tablici 2.11.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32[]	path_x
float32[]	path_y

Tablica 2.5. Struktura ROS2 poruke *PathMsg.msg*

Inicijalizacijom ranije opisanih pretplatnika, ujedno se inicijaliziraju i njihovi pozivi. Nakon pokretanja čvora u odgovarajućim vremenskim trenutcima pozivi dohvaćaju poruke ROS2, obrađuju ih i spremaju u globalne varijable čvora. Obrada poruka podrazumijeva raspakiravanje podataka i spremanje u pogodan oblik za obradu.

Osim inicijaliziranih pretplatnika, inicijaliziraju se globalne varijable i izdavači, čija je uloga objavljivanje poruka pretvorenih podataka pogodnih za vizualizaciju te slanje poruka o ažuriranim polazišnim i odredišnim koordinatama u RViz2 alatu. Nazivi tema za objavljivanje s pripadajućim vrstama poruka pogodnih za vizualizaciju podataka prikazani su u tablici 2.12.

Naziv teme	Vrsta poruke
'coast_pcd_points'	sensor_msgs.PointCloud2
'red_pcd_points'	sensor_msgs.PointCloud2
'yellow_pcd_points'	sensor_msgs.PointCloud2
'green_pcd_points'	sensor_msgs.PointCloud2
'safe_zone_pcd_points'	sensor_msgs.PointCloud2
'path_pcd_points'	sensor_msgs.PointCloud2
'raw_path_pcd_points'	sensor_msgs.PointCloud2
'searched_area_pcd_points'	sensor_msgs.PointCloud2
'start_pose_custom'	PoseStamped
'goal_pose_custom'	PoseStamped
'start_goal_msg_update'	StartGoalMsg

Tablica 2.6. Tablica poruka za objavljivanje sa nazivima tema i vrstama poruka

Kako bi proces pretvorbe i vizualizacije uopće i otpočeo potrebno je pokrenuti objavljanje podataka geografskog prostora iz *map_maker* paketa postupkom opisanim u 2.3.5. Nakon objavljivanja podataka mape geografskog prostora, započinje pretplata na temu *gps_coordinates_coast* te poziv pretplatnika samo jednom dohvaća poruka i sprema podatke o geografskim koordinatama obale i zona udaljenosti u globalne varijable čvora. Ovakav pristup dohvaćanja poruke usvojen je kako bi mapa geografskog prostora bila dostupna vizualizacijskom alatu i u slučaju prekida objavljivanja podataka na temu *gps_coordinates_coast*. S druge strane, poziv ima mogućnost prepoznavanja novih podataka geografskog prostora te se u tom slučaju ponovo inicijaliziraju globalne varijable čvora kako bi se tijekom procesa mogle vizualizirati različite mape geografskog prostora bez ponovnog pokretanja *rviz2_poincloud_launch.py* launch datoteke.

Dohvaćanjem poruke s podacima o geografskom prostoru započinje proces pretvorbe podataka. Prvi je korak procesa spajanje razdvojenih lista geografskih koordinata obale i zona udaljenosti u višedimenzionalna polja. Na osnovi ovog koraka stvara se lokalni koordinatni sustav za što je zadužena klasa *local_coordinates_converter.py* po principima opisnim u poglavlju 2.2. klasa *local_coordinates_converter.py* inicijalizira liste obale i

zona udaljenosti, spaja ih u jednu listu te postavlja rubne koordinate kao minimalne i maksimalne vrijednosti spojene liste. Nadalje, iz rubnih koordinata se stvara lokalni koordinatni sustav te se pomoću funkcija pretvorbe koordinate globalnog koordinatnog sustava pretvaraju u koordinate lokalnog koordinatnog sustava.

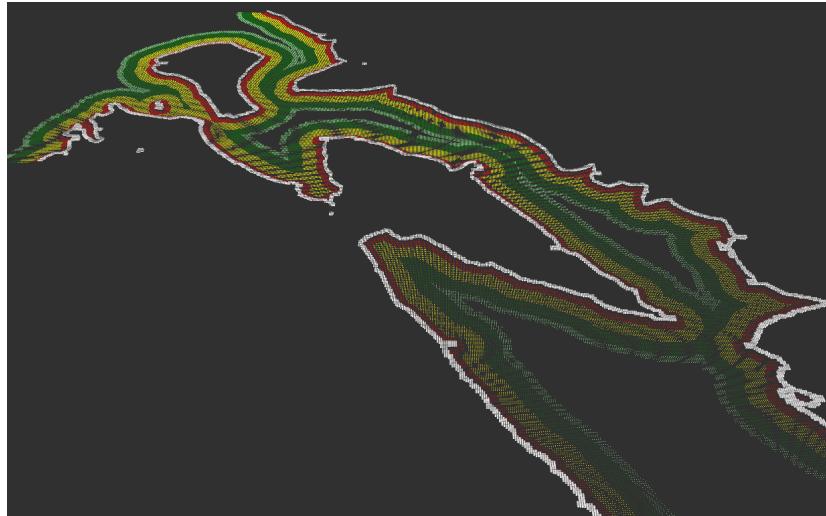
Liste koordinata obale i zona udaljenosti lokalnog koordinatnog nisu pogodne za vizualizaciju u RViz2 alatu te se zato pretvaraju u oblik poruke

sensor_msgs.PointCloud2. Poruka *sensor_msgs.PointCloud2* je standardna poruka koja se koristi za predstavljanje skupa točaka u trodimenzionalnom prostoru prostoru. Obično se koristi za predstavljanje podataka iz senzora kao što su LiDAR-i ili 3D kamere te predstavljanje geografskih koordinata. Svaka točka u oblaku može sadržavati više polja. Polje Fields definira tipove svakog polja u oblaku točaka (Uobičajena polja uključuju 'x', 'y', 'z' i opcionalno 'rgb' i 'intensity'), a polje bigendian koje označava je li format podataka big-endian ili little-endian. Nadalje, polje Point_step naznačuje je li veličina jedne točke u bajtovima, dok Row_step označava veličinu jednog reda u bajtovima. Polje Data sadrži stvarne podatke oblaka točaka serijalizirane u nizu bajtova. Polje Is_dense označava postoje li nevaljane točke u podatcima. Detaljniji se opis sadržaja poruke može pronaći na ROS2 internetskim stranicama.

Proces pretvorbe koordinata lokalnog koordinatnog sustava u poruku *sensor_msgs.PointCloud2* vrši funkcija *convert_to_pcd*. U funkciji je pretvorbe potrebno koordinatama postaviti treću dimenziju *height*. Vrijednost se treće dimenzije postavlja u ovisnosti o preferencijama trodimenzionalnog prikaza koordinata. Koordinate se potom pretvaraju u polje vrste NumPy array te se serijalizira u niz bajtova. Nadalje, inicijalizira se polje fields pomoću Python-ove liste comprehensions za kreiranje liste objekata tipa *sensor_msgs.PointField*. Svaki objekt predstavlja jedno polje u *PointCloud2* poruci, definirajući kako su pojedini atributi (x, y, z) točaka pohranjeni. Na kraju se postavljaju vrijednosti point_step i row_step, izrađuje se *sensor_msgs.PointCloud2* poruka koja se vraća pozivatelju funkcije.

Proces pretvorbe završava spremanjem poruke *sensor_msgs.PointCloud2* u globalne varijable čvora te objavom istih na temama prikazanim u tablici 2.12. Datoteka *textit-map_visualization.rviz* (iz *rviz2* programske mape) sadrži definirane postavke za vizualizaciju pretvorenih podataka pa se objavom na zadane teme podatci automatski pri-

kazuju unutar RViz2 alata. Primjer trodimenzionalnog prikaza koordinata obale i zona udaljenosti unutar RViz2 vizualizacijskog alata nalazi se na slici 2.49.



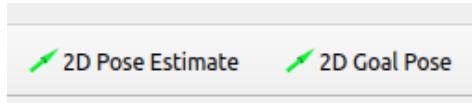
Slika 2.28. Vizualizacija obale i zona udaljenosti u RViz2 vizualizacijskom alatu

Shodno ranije navedenom, u RViz2 vizualizacijskom se okruženju pomoću poruke *sensor_msgs.PointCloud2* vizualiziraju koordinate obale, crvena, žuta, zelena i sigurna zona udaljenosti, koordinate interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje. Kako su navedeni podatci skupovi koordinata, vrijede ista pravila pretvorbe koordinata iz geografskih koordinata u koordinate lokalnog koordinatnog sustava te potom u *sensor_msgs.PointCloud2*. Jedini podatci za koje nije korišten oblak točaka za vizualizaciju jesu koordinate polazišne i odredišne točke. Proces njihovog postavljanja i vizualizacije bit će opisan u narednom poglavlju 2.6.3.

2.4.3. Postavljanje polazišne i odredišne točke te pokretanje ROS2 action client i action server za planiranje putanje plovila

Kako i sam naslov ovog poglavlja naznačuje, glavna uloga paketa *path_planning_client* je postavljanje polazišne i odredišne točke koje će algoritam koristiti za planiranje putanje plovila. Procesom postavljanja točaka putanje upravlja čvor *start_goal_publisher.py* koji pokreće *rviz2_pointcloud_launch.py* launch datoteke zajedno s čvorom pretvorbe podataka za vizualizaciju *pointcloud_publisher.py* te čvorom *rviz_launch.py* koji pokreće RViz2 vizualizacijski alat sa spremlijenim postavkama vizualizacije.

Postavljanje polazišne i odredišne točke putanje je moguće iz dva različita izvora. Prvi je postavljanje točaka u Rviz2 alatu. Naime, Rviz2 alat ima dvije različite opcije u alatnoj traci za objavljuvanje podataka o označenoj poziciji na geografskoj mapi. Sam alat objavljuje pozicije postavljenih točaka na temama na koje se lako pretplatiti, stoga je idealan u svrhu postavljanja odredišne i polazišne točke. Za polazišnu točku odabrana je opcija na alatnoj traci pod imenom *2D pose estimation* koja objavljuje podatke o postavljenoj poziciji na mapi (postavlja se klikom miša na željenu poziciju) na temu *initialpose*. Vrsta poruke objavljene na temi je *geometry_msgs.PoseWithCovarianceStamped* koja se koristi za predstavljanje položaja (pozicije i orientacije) robota ili bilo kojeg objekta u trodimenzionalnom prostoru, zajedno s kovarijacijskom matricom koja označava nesigurnost procjene položaja. U narednim bi fazama projekta umjesto postavljenje pozicije u RViz2 alatu trebala biti izrađena tema na koju se objavljuju informacije sa GPS uređaja stvarnog plovila. Upravo zato je *geometry_msgs.PoseWithCovarianceStamped* korisna za predstavljanje polazišne pozicije putanje jer uključuje kovarijacijsku matricu nesigurnosti određenja pozicije što je karakteristika svakog senzora. Nadalje, prilagodba programa za pretplatu na novo izrađenu temu koja sadrži GPS podatke uređaja umjesto na temu *initialpose* relativno jednostavna jer navedene teme imaju istu vrstu poruke. Za odredišnu je točku putanje odabrana opcija na alatnoj traci pod imenom *2D Goal Pose* koja objavljuje podatke o postavljenoj poziciji na mapi (postavlja se klikom miša na željenu poziciju) na temu *goal_pose*. Vrsta poruke objavljena na temi je *geometry_msgs.PoseStamped* koja za razliku od *geometry_msgs.PoseWithCovarianceStamped* nema uključenu nesigurnost određivanja pozicije što je također željeno vladanje jer odredišna točka, zbog same prirode procesa, ne bi trebala biti postavljena s uključenom nesigurnosti. Na navedene objavljene teme s ažuriranjima polazišne i odredišne točke pretplaćuje se čvor *pointcloud_publisher.py* koji upravlja procesom vizualizacije te sadrži podatke o pozicijama unutar lokalnog koordinatnog sustava. Obrnutim postupkom pretvorbe podataka od postupka u prethodnom poglavlju 2.6.2., koordinate postavljene u Rviz2 alatu iz lokalnog koordinatnog sustava se pretvaraju u geografske koordinate globalnog koordinatnog sustava te putem izdavača objavljuju na temi *start_goal_msg_update* kao *StartGoalMsg.msg* poruku (tablica 2.10.). Dio alatne trake s *2D pose estimation* i *2D goal pose* opcijama prikazan je na slici 2.50.



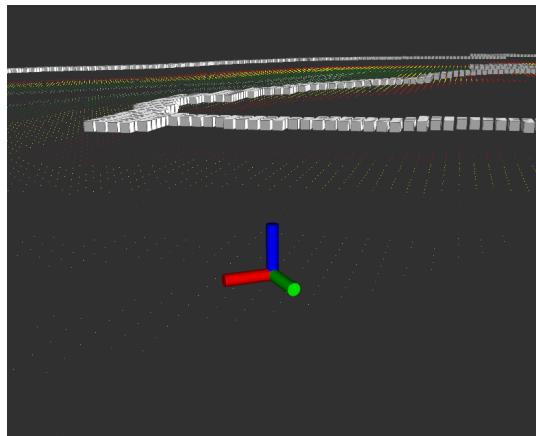
Slika 2.29. Rviz2 alatna traka

Druga je opcija postavljanja polazišne i odredišne točke putem čvora *start_goal_update* koji se pokreće putem *update_start_goal.launch.py* launch datoteke. Launch datoteka ima opciju prosljeđivanja podataka iz Linux terminal naredbe o polazišnoj točki putanje, odredišnoj točki putanje ili obje istovremeno. Navedeni program provjerava valjanost proslijedjenih podataka te ih postavlja u *StartGoalMsg.msg* poruku čiji sadržaj objavljuje na temi *start_goal_msg_update_manual*. Sufiks *manual* naznačuje da su podaci objavljeni "ručno", tj. da ih je korisnik unio putem Linux terminal naredbe. Poruka se objavljuje samo kratak vremenski period dovoljan da *start_goal_publisher.py* dohvati objavljene podatke te ih spremi interno u globalne varijable čvora. Nakon što istekne vremenski period objavljivanja, čvor se gasi kako bi se moglo postaviti nove informacije o polazišnoj i odredišnoj točki putanje bez potrebe za prisilnim prekidanjem programa.

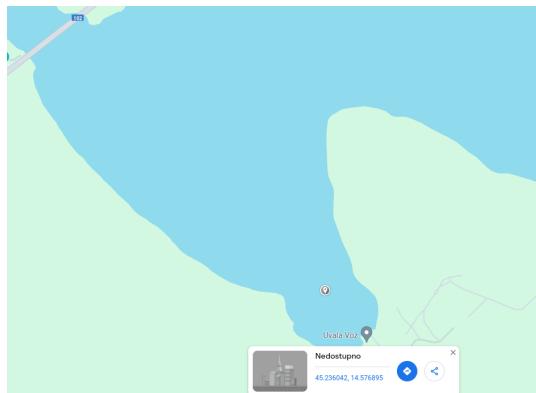
ROS2 čvor *start_goal_publisher.py* zadužen za postavljanje globalne odredišne i polazišne točke putanje se pretplaćuje na *start_goal_msg_update* i *start_goal_msg_update_manual* te osluškuje ažuriranja podataka na navedenim temama. Svakim ažuriranjem vrijednosti na temama, novi podatci se zajedno s vremenskim okvirom postavljaju u globalne varijable čvora. Vremenski je okvir značajan čimbenik procesa jer on određuje aktualnost podataka, odnosno koje su informacije i s koje teme posljednje ažurirane. S obzirom na aktualnost podataka, inicijalizirani je izdavač programa zadužen za objavljivanje globalne početne i polazišne točke, u pripadajućem pozivu objavljuje *StartGoalMsg.msg* poruku na temu *start_goal_msg*. Na temu je pretplaćen i *poincloud_publisher.py* koji geografske koordinate polazišne i odredišne točke globalnog koordinatnog sustava pretvara u koordinate lokalnog koordinatnog sustava te potom u *geometry_msgs.PoseStamped* poruku pogodnu za vizualizaciju u RViz2 alatu. Navedene se poruke objavljuju na teme *start_pose_custom* i *goal_pose_custom*, a potom se vizualiziraju u Rviz2 2 alatu, što je prikazano na slikama 2.55. i 2.56.

Kako bi se dokazala valjanost postavljenih geografskih koordinata putem ažuriranja u Linux terminalnu pomoću *start_goal_update.py* čvora proveden je eksperiment koji uključuje dohvaćanje geografskih podataka s javno dostupne Google Maps internetske stranice. Geografske koordinate polazišne i odredišne točke putanje preuzete su s Google Maps internetske stranice (slike 2.53. i 2.54.) te potom zaliđejene u naredbu za pokretanje *update_start_goal_launch.py* launch datoteke. Ažurirane polazišne i odredišne točke u Rviz2 alatu prikazane na slikama 2.55. i 2.56. odgovarale su geografskim koordinatama iz Google Maps alata, čime je dokazano da sustav ima mogućnost upravljanja stvarnim geografskim koordinatama postavljenim iz drugih izvora podataka.

Vizualizacijom polazišne i odredišne točke u trodimenzionalnom prostoru na prikazanoj geografskoj mapi korisnik može ocijeniti valjanost postavljenih točaka pa ih navedenim postupcima ažurirati. U trenutku kada je korisnik zadovoljan postavljenim točkama pokreće se **ROS2 mehanizam komunikacije action client i action server** koji upravlja procesom planiranja putanje. Mehanizam komunikacije se pokreće putem *start_goal_client_launch.py* launch datoteke koja poziva ROS čvor *start_goal_client.py*. Čvor u konstruktoru inicijalizira pretplatnika na temu *start_goal_msg* sa globalno postavljenom polazišnom i odredišnom točkom putanje. Nadalje inicijalizira izdavače za objave interpolirane putanje i neinterpolirane putanje na teme *path* i *raw_path* te izdavač za objavljivanje područja pretrage algoritma planiranja putanje na temu *searched_area*. Naposljetku se inicijalizira i akcijski klijent *start_goal_client* s akcijom naziva *StartGoalAction.action* iz *user_action_interfaces* paketa. Akcijski klijent je ROS2 entitet koji pokreće dugotrajne zadatke na serveru. Klijent šalje zahtjeve serveru, može ih otkazati i primati povratne informacije i rezultate. Akcijski server i klijent omogućuju asinkronu komunikaciju, što je korisno za dugotrajne zadatke poput planiranja putanja plovila ili bilo kojeg zadatka koji traje duže vrijeme i zahtjeva praćenje napretka. Akcijski klijent i server komuniciraju putem akcije koja ima definiranu strukturu s poljima zahtjev, rezultat i povratna informacija. Polje zahtjev popunjava akcijski klijent te se informacija proslijeđuje akcijskom serveru koji u povratnoj vezi klijentu vraća povratne informacije te naposljetku i dobiveni rezultat. Prikaz sadržaja akcije *StartGoalAction.action* dostupan je u tablici 2.13.



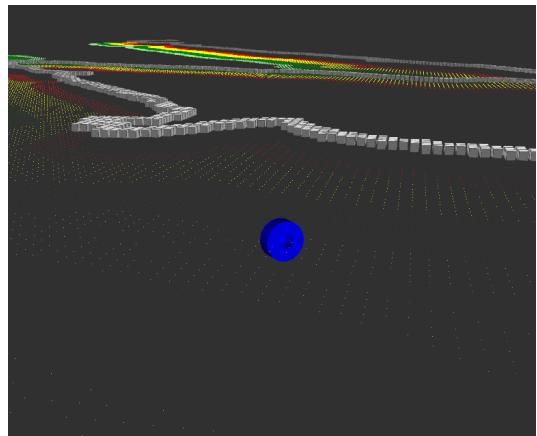
Slika 2.30. Vizualizacija polazišne točke putanje u RViz2 alatu



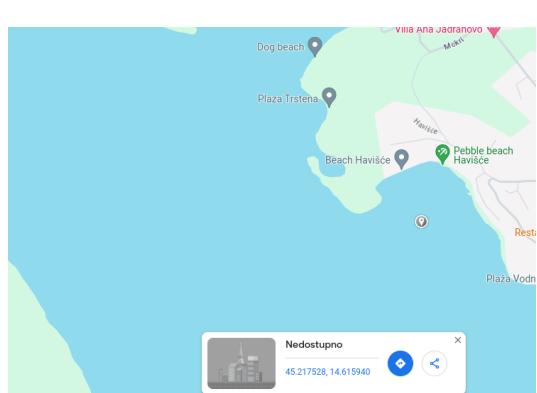
Slika 2.32. Postavljena polazišna točka u Google Maps alatu



Slika 2.34. Vizualizacija polazišne točke iz Google Maps alata u Rviz2 alatu



Slika 2.31. Vizualizacija odredišne točke putanje u RViz2 alatu



Slika 2.33. Postavljena odredišna točka u Google Maps alatu



Slika 2.35. Vizualizacija odredišne točke iz Google Maps alata u Rviz2 alatu

Vrsta Podataka	Naziv	Tip
float32[]	start	zahtjev
float32[]	goal	zahtjev
<hr/>		
float32[]	path_x	rezultat
float32[]	path_y	rezultat
float32[]	raw_path_x	rezultat
float32[]	raw_path_y	rezultat
float32	path_distance	rezultat
float32	raw_path_distance	rezultat
float32	estimated_path_time	rezultat
float32	estimated_raw_path_time	rezultat
<hr/>		
float32[]	partial_path_x	povratna informacija
float32[]	partial_path_y	povratna informacija
float32[]	search_area_x	povratna informacija
float32[]	search_area_y	povratna informacija

Tablica 2.7. Struktura ROS2 akcije *StartGoalAction.action*

Nastavno na prethodno navedeno, nakon inicijalizacije pretplatnika, izdavača i akcijskog klijenta čvor *start_goal_client.py* u pozivu preuzima podatke sa *start_goal.msg* i proslijeduje ih funkciji za slanje zahtjeva akcijskom klijentu. Funkcija potom poziva inicijaliziranu akciju, postavlja polje zahtjev te osluškuje dostupnost servera. Pokretanjem se uspostavlja asinkrona komunikacija te se serveru predaje zahtjev koji sadrži polazišnu i odredišnu točku putanje uz inicijalizaciju funkcija za primanje povratnih informacija i rezultata. Server primitkom zahtjeva započinje s procesom planiranja putanje plovila te u povratnoj petlji vraća informacije u području pretrage algoritma planiranja putanje. Područje pretrage zapravo predstavlja geografski prostor koji je algoritam planiranja putanje istražio do trenutka objavljivanja povratne informacije što je izuzetno korisno za promatranje i ispravljanje rada algoritma. Povratne se informacije spremaju u globalne varijabla čvora i putem izdavača objavljaju na temu *searched_area*. Nadalje, u trenutku nakon što je putanja isplanirana, server vraća u polju rezultat geografske koordinate interpolirane i neinterpoliran putanje uz pripadajuće informacije o duljini putanja i procijenjenom vremenu trajanja plovidbe te ih akcijski klijent sprema u globalne varijable i objavljuje na teme *path,raw_path* i *path_info* čime završava proces planiranja putanje. Putem teme *path_info* dostupne su informacije o duljini putanja i procijenjenom vremenu putem poruke *PathInfMsg.msg* iz *user_action_interfaces* paketa čiji je sadržaj prikazan u tablici 2.14.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32	path_distance
float32	raw_path_distance
float32	estimated_path_time
float32	estimated_raw_path_time

Tablica 2.8. Struktura ROS2 poruke *PathInfoMsg.msg*

Pri tomu važan je čimbenik procesa i čvor *poincloud_publisher.py* koji pretplatom na teme iz *searched_area* omogućuje vizualizaciju područja pretrage algoritma planiranja putanje te naposljetu i isplaniranih putanja. Područje pretrage i isplaniranih putanja se pretvaraju u lokalni koordinatni sustava te potom u poruku *sensor_msgs.PointCloud2* i putem izdavača objavljaju na temama *searched_area_pcd_points*, *raw_path_pcd_points* i *raw_path_pcd_points*. Navedene teme se vizualiziraju u Rviz2 alatu.

Zaključno, u ovome je poglavlju opisan proces vizualizacije geografskih koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, polazišne i krajnje točke putanje, interpolirane i neinterpolirane putanje kao i područja pretrage algoritma planiranja putanje. Također je objašnjen proces postavljanja polazišne i odredišne točke putanje uz action client i action server mehanizam komunikacije za planiranje putanje plovila. Jedino što je preostalo opisati u glavnom dijelu ovog rada je dio čiji naziv je ujedno i naziv rada, a to je **modelski informirano globalno planiranje putanje plovila**. Mehanizmi opisani u prethodnim poglavljima izrade mape geografskog prostora i postavljanja polazišne i odredišne točke putanje uz vizualizaciju rezultata su neizostavne sastavnice procesa planiranja putanje bez čijeg razumijevanja i implementacije sam proces nije moguć. Zbog ranije navedenog, u narednom će poglavlju biti opisan modelski informirano globalno planiranje putanje plovila u čijoj je osnovi mapa geografskih koordinata globalnog koordinatnog sustava te pripadajuća polazišna i odredišna točka putanje.

2.5. Globalno planiranje putanje plovila bazirano na mapi cijena prostora i process interpolacije putanje

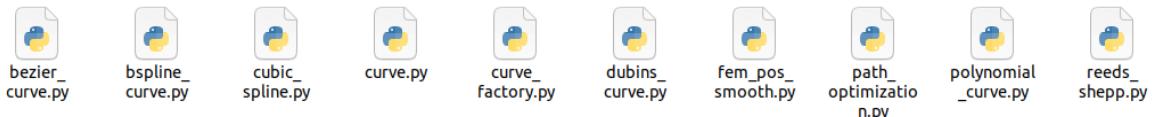
Modelske informirane globalne planirane putanje plovila (engl. Model-based path planning) se odnosi na metode planiranja putanja koje koriste model okruženja i dinamike plovila kako bi odredile optimalnu i efikasnu globalnu putanju plovila po različitim kriterijima. Matematički model okruženja uključuje geografsku mapu plovnog područja s istaknutim koordinatama obale koje označuju prepreku, koordinatama udaljenosti od obale, dubinama mora, preprekama na moru u što se ubrajaju manji otoci i hridi te koordinatama polazišne i odredišne točke putanje. Na osnovi se geografske mape stvara mapa cijena prostora koja predstavlja okolinu u obliku rešetke (grid) gdje svaka komponenta ima pridruženu cijenu koja označava trošak prolaska kroz tu komponentu. Ove cijene mogu biti temeljene na različitim kriterijima kao što su udaljenost od obale, dubina mora ili drugi faktori koji utječu na sigurnost i efikasnost kretanja plovila. Uloga je mape cijena generiranje optimalne putanje plovila u smislu udaljenosti, sigurnosti ili potrošnje energije, a osnovna mu je zadaća izbjegavanje sudara s obalom i preprekama na moru. Mapa cijena izrađuje se pomoću geografskih koordinata prostora dostupnih na temi *gps_coordinates_coast* koju objavljuje čvor *publish_map_launch.py* pretvorbom navedenih koordinata u lokalni koordinati sustav te postavljanjem vrijednosti cijena. Nadalje, bitni čimbenici procesa planiranja putanje su i polazišna i odredišna točka koju serverski čvor opisan u ovome poglavlju prima od klijentskog čvora iz *path_planning_client* paketa putem action client i action server mehanizma komunikacijske. Klijentski čvor nosi naziv *start_goal_client.py*, a serverski čvor *path_planning_server.py* te su oba čvora preplaćena na akciju naziva *StartGoalAction.action*. Pomoću mape cijena i primljene polazišne i odredišne točke putanje, algoritam D* lite računa optimalnu putanju plovila. Izgled konačne putanje ovisi o vrijednostima cijena različitih područja što omogućuje optimizaciju putanje po različitim parametrima. Navedena putanje bez dinamičkog modela nije glatka te posljedično i fizički izvediva. Kako bi putanje koju je generirao D* lite postala fizički izvediva, koriste se tehnike interpolacije koje uključuju postupak raspoređivanja točaka i tehnike prilagođavanja krivulje. Na posljeku serverski čvor vraća akcijskom čvoru putanje u interpoliranom i ne interpoliranom obliku te informacije o putanjama, čime završava proces planiranja putanje plovila.

2.5.1. Struktura i pokretanje ROS2 paketa path_planning_server

Ros2 paket *path_planning_server* je relativno jednostavne strukture u usporedbi s paketima *map_maker* i *path_planning_client* jer sadrži samo jedan serverski čvor *path_planning_server.py*, program *Node.py* koji koristi D* lite algoritam prilikom planiranja putanje i mapu *curve_generation* s programima za interpolaciju putanje. Sama jednostavnost strukture paketa ne treba zavarati jer navedeni paket obavlja veoma složene zadaće u što se ubraja stvaranja lokalnog koordinatnog sustava i mape cijena prostora. Nadalje, u trenutku kada se stvori mapa cijena prostora započinje proces planiranja putanje plovila temeljen na mapi cijena te potom proces interpolacije putanje. Serverski čvor *path_planning_server.py* pokreće launch datoteka *path_planning_server_launch.py* koja definira nekoliko parametara koji upravljaju procesom planiranja i interpolacije putanje te vizualizacijom ulaznih i izlaznih podataka, a moguće ih je mijenjati putem Linux terminala. Parametri koji utječu na planiranje putanje, odnosno na proces izrade mape cijena jesu polje cijena *coast_values_arg* koje definira cijene crvene, žute, zelene i sigurne zone te polje koraka *step_sizes_arg* koje definira korak algoritma u svakoj zoni. Navedeni su parametri neizostavni čimbenici u procesu planiranja putanje jer utječu na brzinu izvedbe algoritma i proces planiranja putanje što je detaljnije opisano u poglavљu 2.6.5. Uz prethodno navedene parametre, serverski čvor prilikom pokretanja launch datoteke inicijalizira pretplatnika na temu *gps_coordinates_coast* s pripadajućim pozivom te ostale globalne varijable koje se koriste tijekom procesa planiranja putanje.

Kako bi proces stvaranja lokalnog koordinatnog sustava otpočeo, kao i kod procesa pretvorbe podataka pogodnih za vizualizaciju 2.6.2. potrebno je pokrenuti launch datoteku *publish_map_launch.py*. Potom pretplatnik serverskog čvora na temu *gps_coordinates_coast* dohvaca u pozivu sadržaj poruke potom se se stvara lokalni koordinatni sustav. Na osnovi se lokalnog koordinatnog sustava izrađuje mapa cijena u obliku rječnika, postavljaju se globalne varijable zatim se inicijalizira komunikacija s akcijskim klijentom. U osnovi serverskog čvora je action client i action server mehanizam komunikacije te je shodno navedenom potrebno pokrenuti klijentski čvor kako bi se proces planiranja putanje nastavio. Nakon pokretanja, serverski čvor uspostavlja putem akcije *StartGoalAction.action* komunikaciju s akcijskim klijentom *start_goal_client.py* koji serveru u polju zahtjev navedene akcije predaje geografske koordinate polazišne i odredišne

točke putanje koje se spremaju u globalne varijable čvora. Nakon izrade mape cijena i inicijalizacije polazišne i odredišne točke putanje, poziva se algoritam koji računa optimalnu putanju s obzirom na vrijednosti diskretiziranog prostora cijena i postavljenih točaka putanje. Detaljni opis izvedbe algoritma planiranja putanje plovila temeljenog na mapi cijena prostora opisan je u poglavljiju 2.5.3. Dobivena putanja uz pomoć navedenog algoritma nije fizički izvediva bez izrade dinamičkog modela plovila koji bi se uključio u proces postavljanja mape cijena. Kako bi se nepostojanje dinamičkog modela nadomjestilo te izradilo univerzalan algoritam planiranja putanje plovila, izrađen je algoritam interpolacije putanje koji uključuje raspoređivanje točaka i tehnike prilagođavanja krivulje. Procesom interpolacije krivulje predsjeda program *path_optimization.py* iz programske mape *curve_generation*. Karakteristike interpolirane putanje također ovise o parametrima predanim u launch datoteci *sampling_rate* koji definira koliko će točaka neinterpolirane putanje algoritam raspoređivana točaka te *optimization_method* koji definira jednu od šest dostupnih metoda zaglađenja krivulje. Programska mapa *curve_generation* prikazana je na slici 2.36.



Slika 2.36. Struktura programske mape *curve_generation*

Parametri vizualizacije launch datoteke definiraju hoće li se pojedini koraci procesa planiranja putanje vizualizirati što je korisno tijekom procesa izrade i ispravljanja grešaka programa. Parametar *show_debug* definira hoće li se vizualizirati ulazni podaci, a parametar *show_results* ulazni podaci s isplaniranim putanjama. Nadalje, parametar *show_feedback* definira hoće li se u povratnoj petlji action client i action server mehanizma komunikacije slati povratne informacije što višestruko usporava proces planiranja putanje, ali pruža uvid u područje pretrage algoritma. Posljednji parametar je polje ograničenja brzina u pojedinim zonama *speed_limits* pomoću kojeg se predviđa trajanje putanje od polazišne do odredišne točke.

Naposljetu, čvor *path_planning_server_launch.py* sprema dobivene koordinate interpolirane i neinterpolirane putanje u globalne varijable serverskog čvora. Koordinate

se potom pozivanjem funkcije pretvorbe pretvaraju iz lokalnog koordinatnog sustava u koordinate globalnog koordinatnog sustava. Također se izračunavaju vrijednosti duljine putanje u metrima i procijenjenog vremena trajanja plovidbe. Koordinate interpolirane i neinterpolirane putanje s pripadajućim duljinama i procijenjenim vremenima trajanja postavljaju se u polje rezultat *StartGoalAction.action* akcije te se prosljeđuju akcijskom klijentu *start_goal_client_launch.py*, čime završava proces planiranja putanje.

Grafički prikaz arhitekture komunikacije ROS2 serverskog čvorova, mehanizma akcijskog klijenta i server te obrade i pretvorbe podataka uz navedene naredbe za pokretanje paketa *path_planning_server* prikazan je na slici 2.37.

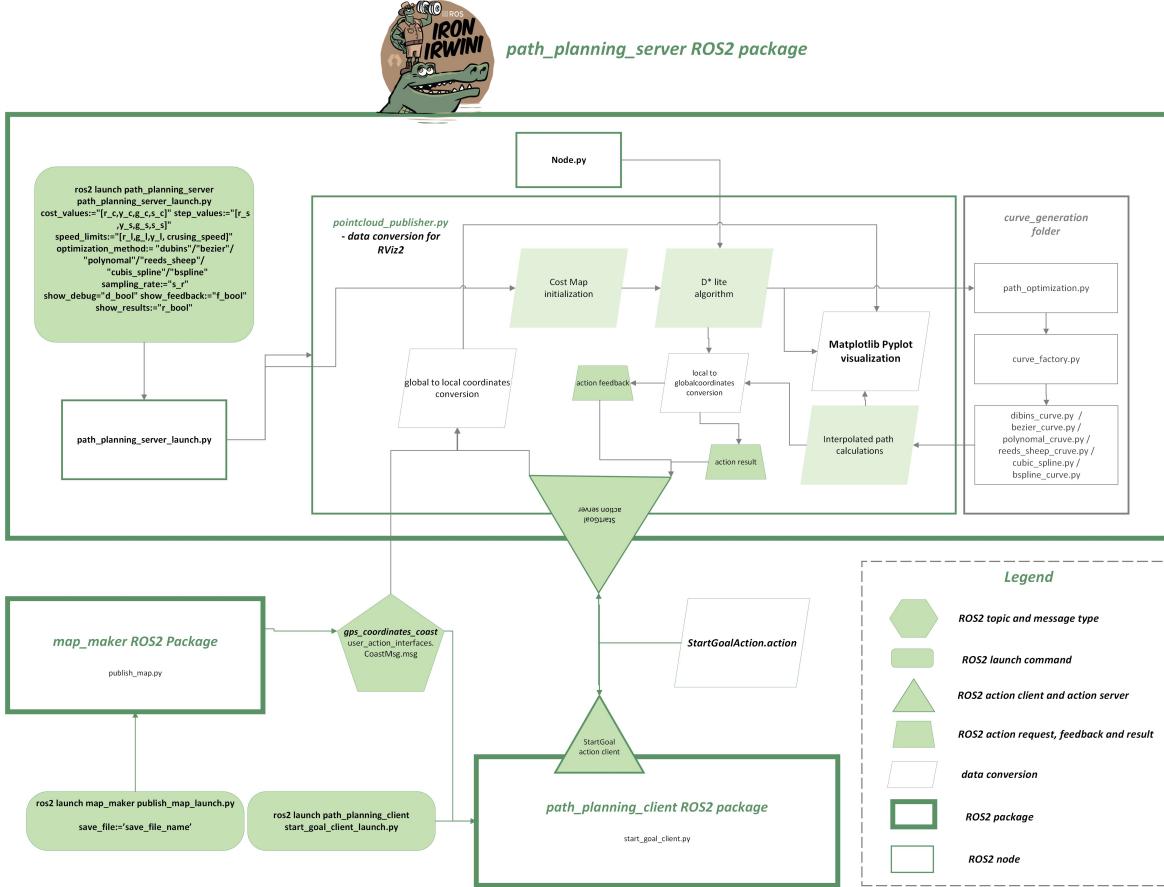
Popis ROS2 naredbi za pokretanje ROS2 paketa path_planning_server

Jedina zadaća ROS2 paketa *path_planning_server* je planiranje željene putanje plovila pa shodno tome postoji samo jedna naredba u Linux terminalu za pokretanje launch datoteke *path_planning_server_launch.py*. Launch datoteka definira nekoliko parametara za upravljanje procesom izrade mape cijena, interpolacije putanje i vizualizacijom ulaznih i izlaznih podataka. Uloga je navedenih parametara opisana prethodno u ovome poglavlju, ali će u narednim poglavljima biti opisan njihov utjecaj na navedene procese. Naredba glasi:

```
ros2 launch path_planning_server path_planning_server_launch.py  
cost_values:=[r_c,y_c,g_c,s_c]  step_values:=[r_s,y_s,g_s,s_s]  
speed_limits:=[r_l,g_l,y_l, crusing_speed] optimization_method:=  
"dubins"/"bezier"/... sampling_rate:="s_r" show_debug="d_bool"  
show_feedback:="f_bool" show_results:="r_bool"
```

Gdje su unaprijed zadane vrijednost u launch datoteci:

```
[r_c,y_c,g_c,s_c] := [10.0,2.0,1.5,1.2]  
[r_s,y_s,g_s,s_s] := [50.0,50.0,100.0,100.0]  
[r_l,g_l,y_l, crusing_speed] := [2.0,5.0,8.0,25.0]  
optimization_method:= "dubins" s_r := 5.0  
d_bool == f_bool == r_bool := False
```



Slika 2.37. Arhitektura komunikacije ROS2 paketa `path_planning_server`

2.5.2. Testiranje algoritama globalnog planiranja putanje

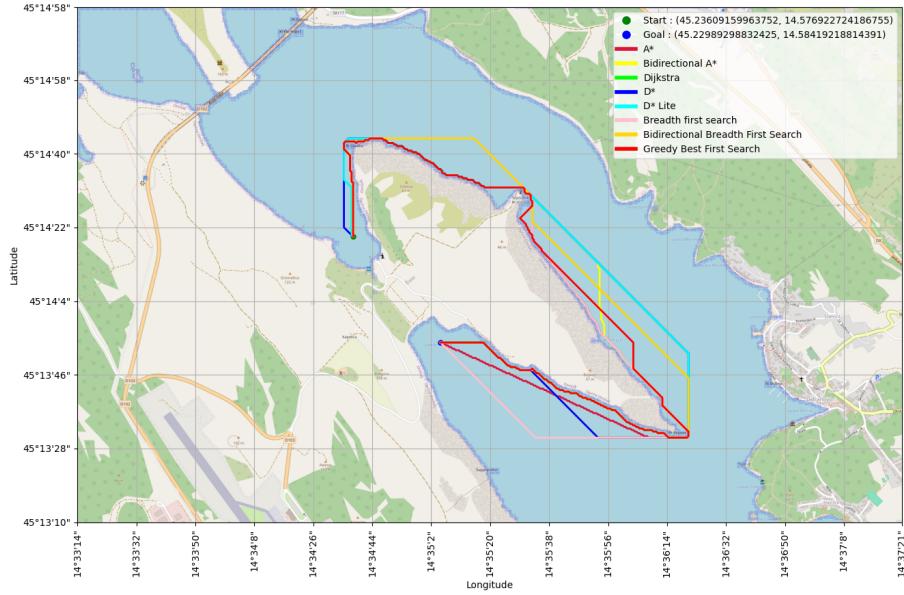
Prema definiciji globalno je planiranje puta opsežna izvanmrežna (offline) metoda planiranja optimalne putanje koja se temelji na pruženim informacijama o plovidbenom potpodručju putem geografskih mapa te ne zahtijeva rad u realnom vremenu, što je detaljno opisano u poglavlju 1.3. U navedenom je poglavlju dano obrazloženje zašto se u ovome radu za proces planiranja putanje koriste klasični algoritmi globalnog planiranja putanje. Međutim, potrebno je ispitati koji od klasičnih algoritama planiranja putanje je optimalan za zadaću planiranja putanje plovila s obzirom na različite kriterije optimalnosti. U ovome je poglavlju, također, ispitano nekoliko klasičnih algoritama planiranja putanje dostupnih u GitHub repozitoriju PythonRobotics [2]. Navedeni algoritmi iz repozitorija prilagođeni su za rad s binarnom mapom prostora, odnosno kao ulazne podatke primaju informacije o preprekama (obali) te polazišnoj i odredišnoj točki putanje. Prilagodba bi svih algoritama za rad s mapom cijena bila vremenski zahtjevna te shodno tome ovaj korak nije obrađen u okviru ovoga rada. Kao kriterij optimalnosti

algoritama uzeti su duljina putanje i vrijeme izvršavanja na biranoj mapi prostora. Također, razmotrene su prilagodbe algoritama procesa lokalnog planiranja putanje, odnosno dinamičkim okruženjima. Mapa prostora za testiranje algoritama izrađena je na principu opisanom u poglavlju 2.3. s parametrom uzorkovanja 10.

Ispitani su klasični algoritmi A*, dvosmjerni A*, Dijkstra, D*, D* Lite, pretragu u širinu (BFS), dvosmjernu pretragu u širinu i pretragu najboljim prvim pokušajem (GBFS). A* je algoritam koji kombinira trošak puta od početnog do trenutnog čvora s heurističkom procjenom preostalog troška do cilja, čime osigurava optimalne i efikasne putanje u mnogim slučajevima. Dvosmjerni A* unapređuje ovu metodologiju pretražujući istovremeno od početka i cilja, što može značajno smanjiti vrijeme pretrage. Dijkstra algoritam, s druge strane, koristi uniformni trošak pretrage i optimalan je, ali može biti manje efikasan od A* u velikim prostorima zbog nedostatka heurističke smjernice. D* i D* Lite su algoritmi optimizirani za statičke i dinamičke prilike, gdje se uvjeti puta mogu mijenjati, omogućujući efikasno ažuriranje već pronađenih putanja bez ponovnog računanja od nule. Pretraga u širinu (BFS) je jednostavan algoritam koji istražuje sve čvorove na trenutnoj dubini prije nego što li prijeđe na sljedeću razinu, što ga čini pogodnim za neinformirane pretrage, dok dvosmjerna BFS pretraga unapređuje ovu metodologiju pretražujući od početne i ciljne tačke istovremeno. Greedy best-first search (GBFS) koristi heurističku funkciju za pretraživanje, ali se fokusira isključivo na minimiziranje procijenjenog preostalog troška do cilja, što može dovesti do neoptimalnih putanja. Ono što je poznato prije testiranja, u pogledu duljine putanje je da algoritmi A*, bidirectional A*, Dijkstra, D* i D* Lite osiguravaju najkraću putanju, ako se koriste u odgovarajućim uvjetima. BFS i njegove varijante ne garantiraju najkraću u svim uvjetima, dok GBFS nije optimalan za najkraću putanju. Na temelju navedenog, postavljena je pretpostavka da duljine putanja u velikom broju slučajeva malo odstupaju ili su jednake zbog specifičnih mehanizama na kojima se temelji, osim duljine putanje GBFS algoritma te ponekad i BFS algoritma. Međutim, pretpostavku je bilo potrebno ispitati na mapi plovног područja geografskog prostora.

Ispitivanje je provedeno na mapi prostora različitih područja otoka Krka s karakterističnim geografskim obilježjima poluotoka, zaljeva i uvala. Na slici 2.38. je prikazano planiranje putanje plovila iz uvale Voz do uvale Peškera koje se nalaze na suprotnim stra-

nama poluotoka. Na slici 2.40. je prikazano planiranje putanje iz luke terminala Omišalj (Omišaljski zaljev) do nasumično postavljene točke otvorenog mora na približno jednakoj geografskoj dužini. Na slici 2.42. je prikazano je planiranje putanje iz zaljeva Soline kojeg karakterizira uski tjesnac kod rta Sulinj do uvale Jazbina. Rezultati izgleda isplaniranih putanja prikazani su na navedenim slikama, a pripadajuće tablice s rezultatima kriterija optimizacije na slikama 2.39., 2.41. i 2.43.

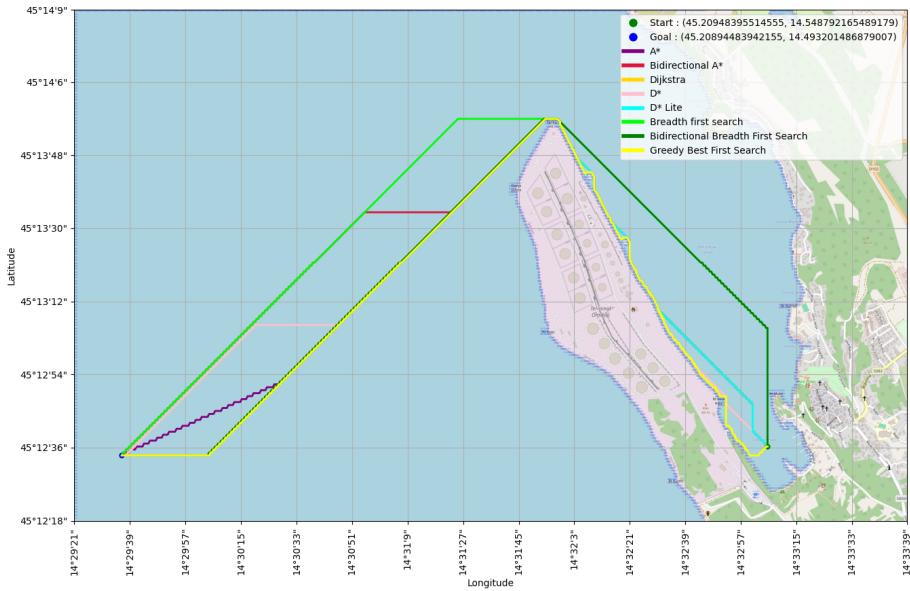


Slika 2.38. Grafički prikaz rezultata planiranje putanje plovila iz uvale Voz do uvale Peškera

	Distance	Time
A*	5898.132752230963	102.53748
Bidirectional A*	5898.132752230956	102.55466
Dijkstra	5898.132752230963	102.47737
D*	5888.132752230956	39.82906
D* Lite	5898.132752230963	41.4521
Breadth first search	5898.132752230965	104.41701
Bidirectional Breadth First Search	5898.132752230957	100.90167
Greedy Best First Search	5998.549057834391	105.29984

Slika 2.39. Tablica duljina putanja i vremena izvršavanja algoritama

Iz prikazanih je tablica vidljivo da su vremena izvršavanja algoritama relativno velika. Razlog tome je što algoritmi nisu optimizirani za rad na mapi većih skala, međutim informacije su korisne prilikom usporedbe rezultata. Također je vidljivo da rezultati vremena izvršavanja programa ovise o mapi prostora. Naime, u tablici 2.39. najmanje vrijeme izvršavanja ima D* algoritma te potom D* lite algoritam, iako razlika nije velika, dok ostali algoritmi imaju više od duplo veće vrijeme izvršavanja. Razlog tome se nalazi

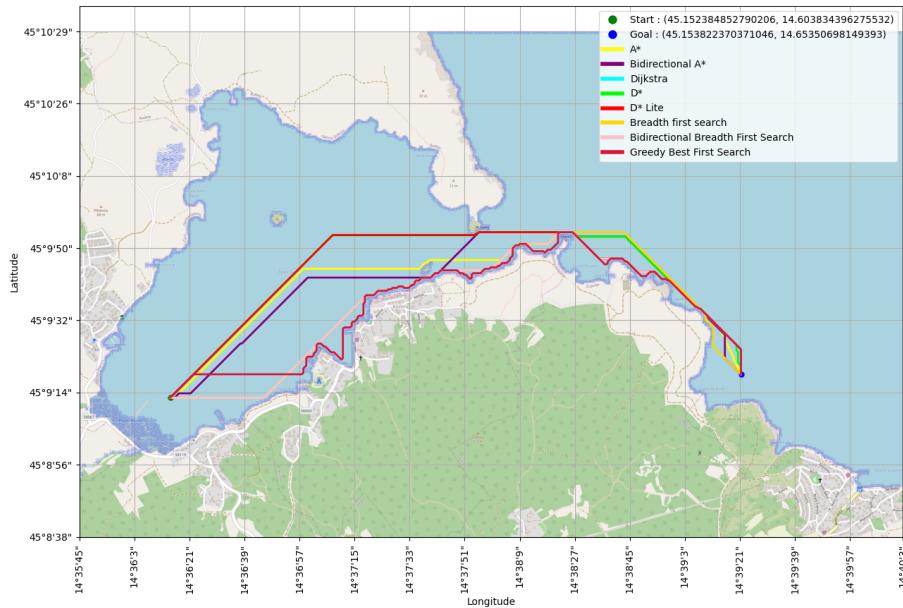


Slika 2.40. Grafički prikaz rezultata planiranje od luke Omišalj do točke otvorenog mora

	Distance	Time
A*	6678.448045156698	80.04693
Bidirectional A*	6678.4480451566815	84.2569
Dijkstra	6678.448045156699	79.94543
D*	6668.4480451566915	73.57965
D* Lite	6678.448045156699	75.32312
Breadth first search	6678.448045156697	78.25185
Bidirectional Breadth First Search	6678.448045156697	78.79695
Greedy Best First Search	6943.889603929548	76.44094

Slika 2.41. Tablica duljina putanja i vremena izvršavanja algoritama

u činjenici da D* i D* Lite mogu efikasnije upravljati memorijskim i računalnim resursima zbog svoje sposobnosti fokusiranja pretrage na relevantne dijelove mape. Zato algoritmi bolje funkcioniraju u složenim okruženjima kao što je planiranje putanje iz jedne u drugu uvalu na suprotnim stranama poluotoka kroz uži morski kanal s razvedenom obalom što je prikazano na slici 2.38. Međutim, u tablici 2.41. vremena izvršavanja algoritama su mjerljiva, odnosno njihova razlika nije velika. Područje planiranja putanje sa slike 2.40. nije zahtjevno kao područje s prethodno opisane slike jer većinu mape čini otvoreni prostor mora s ne toliko razvedenom obalom, dok je Omišaljski zaljev podosta širok. U takvom okruženju vremena izvršavanja nisu previše različita jer se radi o većinom otvorenom prostoru pretrage bez velikog broja prepeka. Iz rezultata tablice 2.43. je dobiven isti zaključak kao iz tablice 2.39. da u složenijem prostoru kao što je zaljev zatvorenog tipa s uskim prolazom na izlazu iz zaljeva najbolje rezultate vremena izvr-



Slika 2.42. Grafički prikaz rezultata planiranje od zaljeva Soline do uvale Jazbin

	Distance	Time
A*	4770.437226013952	118.09266
Bidirectional A*	4770.437226013957	117.86109
Dijkstra	4770.437226013951	115.77486
D*	4760.437226013956	39.70043
D* Lite	4770.437226013951	34.97116
Breadth first search	4770.437226013969	117.11278
Bidirectional Breadth First Search	4770.437226013961	119.67495
Greedy Best First Search	5390.437226013958	114.99232

Slika 2.43. Tablica duljina putanja i vremena izvršavanja algoritama

šavanja polučuju D* i D* lite algoritam, dok su vremena izvršavanja ostalih algoritama višestruko veća. Po kriteriju optimalnosti vremena izvršavanja algoritmi D* i D* lite su se kroz testiranje u različitim geografskim mapama pokazali kao optimalni, dok su ostali algoritmi u složenijim mapama imali višestruko veće vrijeme izvršavanja te se zato odbacuju.

Iz tablica je vidljivo da duljine isplaniranih putanja ne variraju u ovisnosti geografskim prostorima kao vremena izvršavanja te algoritmi A*, bidirectional A*, Dijkstra, D*, D* Lite, BFS i bidirectional BFS imaju gotovo jednake rezultate. Jedini algoritam koji se ističe po kriteriju duljine je GBFS čime je potvrđena ranija pretpostavka. Shodno navedenom, osim GBFS svi su algoritmi optimalni po kriteriju duljine isplanirane putanje.

Osim kriterija optimalnosti po duljini putanje i vremenu izvršavanja algoritma, kao kriterij optimalnosti je uzeta i mogućnost prilagodbe algoritama lokalnom planiranju putanje, odnosno dinamičkim okruženjima. Od navedenih algoritama D* i D* Lite su dizajnirani s namjenom planiranja putanje u dinamičkim okruženjima, omogućujući brzo ponovno planiranje putanja kada se uvjeti promijene. A* i njegovi varijanti, poput dvosmjernog A*, također se mogu prilagoditi za dinamička okruženja uz određene modifikacije, iako nisu izvorno razvijeni za te uvjete. Dijkstra algoritam, poznat po svojoj optimalnosti, može se koristiti u dinamičkim okruženjima, ali može biti manje efikasan zbog potrebe za ponovnim računanjem cijele putanje pri svakoj promjeni. Pretraga u širinu (BFS) i njezina dvosmjerna varijanta te pretraga najboljim prvim putem (GBFS) manje su pogodniji za dinamička okruženja zbog svoje neoptimalnosti i neefikasnosti u ponovnom planiranju putanja. Osim po kriteriju optimalnosti vremena izvršavanja, algoritmi D* i D* lite optimalni su i po kriteriju prilagodbe algoritma dinamičkim okruženjima. Potrebno je naglasiti da glatkoća i izvedivost putanje nisu bili kriterij optimalnosti prilikom odabira algoritma iako su izgledi putanja prikazani na slikama jer je teško postaviti u perspektivu izgled putanje algoritama na mapi cijena prostora bez prilagodbe algoritama što je vremenski zahtjevan proces.

Kroz ispitivanje kriterija oba su se algoritma D* i D* lite pokazala kao optimalni, ali je naposljetku odlučeno da će algoritam D* lite biti korišten kao glavni algoritam modelski informiranog globalnog planiranja putanje. Razlog tomu je činjenica da navedeni algoritam ima jednostavniju implementaciju u usporedbi s D* algoritmom, a što je važno kod prilagodbe algoritma mapi cijena. Nadalje, D* Lite je specifično optimiziran za rad u dinamičkim okruženjima. Njegova arhitektura omogućava efikasnu obradu promjena u stvarnom vremenu, prilagođavajući se novim uvjetima bez potrebe za potpunim ponovnim izračunom putanje. S druge strane, u dinamičkim okruženjima kod promjene postavki prostora D* algoritam mora ponovno izračunati putanju, što ga čini manje efikasnim od D* lite algoritma. Zaključno, D* lite je odabran zbog jednostavnije implementacije, manjeg utroška računalnih resursa i bolje prilagodbe za rad u dinamičkim okruženjima u odnosu na D* algoritam.

2.5.3. Postupak postavljanja mape cijena i prilagodba D* lite algoritma za modelski informirano globalno planiranje putanje plovila

Procesom modelski informiranog planiranja putanje upravlja ROS2 serverski čvor *path_planning_server.py* koji se pokreće putem launch datoteke *path_planning_server.launch.py* i definira nekoliko parametara koji upravljaju procesom planiranja i interpolacije putanje te vizualizacijom ulaznih i izlaznih podataka, a moguće ih je mijenjati putem Linux terminal naredbe. Parametri koji utječu na planiranje putanje, odnosno na proces izrade mape cijena jesu polje cijena *coast_values* koje definira cijene crvene, žute, zelene i sigurne zone te polje koraka *step_sizes* koje definira korak algoritma u svakoj zoni. Procesom interpolacije upravljaju parametri *sampling_rate* koji definiraju koliko će točaka neinterpolirane putanje algoritam raspoređivanja točaka te *optimization_method* pomoću kojeg se odabire jedna od šest dostupnih metoda zaglađenja krivulje. Nadalje, parametri vizualizacije su parametar *show_debug* za prikaz ulaznih podataka, a parametar *show_results* za prikaz ulaznih podataka s isplaniranim putanjama. Parametar *show_feedback* definira hoće li se u povratnoj petlji action client i action server mehanizma komunikacije slati povratne informacije što višestruko usporava proces planiranja putanje, ali pruža uvid u područje pretrage algoritma. Posljednji parametar je polje ograničenja brzina u pojedinim zonama *speed_limits* pomoću kojeg se previđa trajanje putanje od polazišne do odredišne točke.

U osnovi je procesa planiranja putanje action client i action server mehanizam komunikacije koji predsjeda procesom putem akcije *StartGoalAction.action*. Navedena akcija u polju zahtjev sadrži informacije o geografskim koordinatama polazišne i odredišne točke putanje plovila koju akcijski klijent dostavlja akcijskom serveru. Akcijski server tijekom procesa ima mogućnost slanja povratnih informacija u povratnoj petlji o području pretrage algoritma planiranja putanje, a na kraju procesa u polju rezultati akcijskom klijentu vraća geografske koordinate interpolirane i neinterpolirane putanje zajedno s pripadajućim informacijama o duljini putanje i predviđenom vremenu trajanja plovidbe. Međutim, prije nego li se pristupi procesu planiranja putanje potrebno je izgraditi prostor računanja algoritma.

Kao što je opisano u poglavlju 2.5.2., u ovom su radu ispitani klasični algoritmi planiranja putanje koji koriste mrežu (grid) za reprezentaciju prostora pretraživanja. Ovi algoritmi dijele prostor na diskretne komponente ili čvorove te pretražuju putanju između njih. Jednostavna reprezentacija diskretnih komponenata jesu binarne mape gdje jedna vrijednost predstavlja prepreke, a druga vrijednost slobodni prostor računanja. Proces planiranja putanje plovila je relativno složen te zato zahtjeva detaljniju reprezentaciju prostora od one binarne. Zbog prethodno navedenog, algoritam planiranja putanje u osnovi ima reprezentaciju prostora u obliku mape cijena koja predstavlja okolinu u obliku rešetke (grid) gdje svaka komponenta ima pridruženu cijenu koja označava trošak prolaska kroz tu komponentu. Tijekom pokretanja launch datoteke, program osim parametara iz datoteke u konstruktoru inicijalizira preplatu na temu *gps_coordinates_coast* s pripadajućim pozivom koja u *PathMsg.msg* poruci sadrži informacije o geografskim koordinatama obale i zonama udaljenosti. Također se inicijaliziraju i globalne varijable za spremanje rezultata i među rezultata procesa.

Izrada mape cijena geografskog prostora

Proces planiranja putanje plovila započinje objavu geografskih podataka o koordinatama obale i zona udaljenosti od obale na temu *gps_coordinates_coast* jer poziv pretplatnika tada mijenja zastavicu o primitku podataka. Bez promjene navedene zastavice serverskog čvora poziv samo osluškuje navedenu temu s obzirom na to da bez geografskih podatak nije moguća izrada lokalnog koordinatnog sustava te posljedično i mape cijena prostora. Primitkom podataka sa sadržajem poruke *PathMsg.msg* prvo se razdvojeni podaci o geografskim širinama i dužinama spajaju u višedimenzionalne liste geografskih koordinata. Na principu iz poglavlja 2.2. koji je opisan nekolicinu puta kroz ovaj rad, tvori se lokalni koordinatni sustav pomoću informacija o maksimalnim i minimalnim vrijednostima geografskih koordinata. Geografske koordinate obale i zona udaljenosti od obale se potom iz globalnog koordinatnog sustava pretvaraju u koordinate lokalnog koordinatnog sustava što je preduvjet stvaranja mape cijena prostora. Mapa cijena prostora predstavljena je u obliku rječnika (eng.dictionary) gdje su ključevi koordinate lokalnog koordinatnog sustava, a vrijednost je lista s dva polja. Prvo polje liste je cijena prolaska, a druga je vrijednost koraka algoritma. Polje cijena i koraka koordinata obale nije moguće mijenjati jer obala predstavlja prepreku pa je zato cijena komponente rešetke postavljena

na beskonačno, a korak algoritma je jedan. Vrijednost cijene beskonačno kazuje da algoritam nikada neće prolaziti kroz prepreku. Vrijednosti cijena i koraka zona postavljaju se putem launch datoteke u *cost_values* i *step_sizes* parametrima pa je postavljanjem različitih vrijednost navedenih parametara algoritam moguće optimizirati po različitim kriterijima. U procesu izrade rječnika izrazito je bitno postaviti koordinate obale posljednje jer se u procesu stvaranja lokalnog koordinatnog sustava određene koordinate obale i crvene zone preklapaju. Ukoliko se prvo postave vrijednosti obale te potom crvene zone nastat će praznine u obali zbog kojih algoritam u određenim slučajevima može područje pretrage preliti na prostor kopna. Takvo što nije željeno ponašanje jer izrazito usporava algoritam te krajnja putanja u nekim slučajevima može ići i kroz samo kopno. Osim izrade rječnika (mapa) cijena izrađuju se i rječnici za grafički prikaz područja u kojima su ključevi također geografske koordinate ili koordinate lokalnog koordinatnog sustava, a vrijednosti "r" (red), "y"(yellow),"g" (green", "s"(safe) i "c" (coast). Navedeni rječnici se nakon izrade spremaju u globalne variable serverskog čvora. Na kraju poziva se također inicijaliziraju globalne varijable i funkcije koje algoritam D* lite koristiti prilikom računanja putanje plovila. U narednim koracima projekta u mapu cijena bi trebale biti uključene informacije karakteristične za pomorske mape, a to su dubina i sigurnosne prepreke poput hridi, malih otoka ili postavljenih objekata za signalizaciju prije nego li algoritam može biti testiran u stvarnome okruženju.

Prilagodba D* lite algoritma za globalno planiranje putanje plovila

U trenutku kada su sve vrijednosti inicijalizirane uz izrađenu mapu cijena prostora, u pozivu se inicijalizira funkcija koja uspostavlja komunikaciju s akcijskim klijentom putem action client i action server mehanizma komunikacije. Inicijalizacija navedene funkcija podrazumijeva postavljanje vrijednosti imena akcije *StartGoalAction*, imena klijenta s kojim će server komunicirati *start_goal_client* te imena serverske funkcije poziva. Pokretanjem akcijskog klijenta s navedenim nazivom, poziva se serverska funkcija poziva koja potom inicijalizira primljene geografske koordinate polazišne i odredišne točke putanje iz polja zahtjev te klijentu šalje potvrdu o primitku podataka. Provjerava se valjanost podataka te ukoliko su primljene koordinate valjane pretvaraju se iz globalnog u lokalni koordinatni sustav čime završava proces inicijalizacije i započinje proces planiranja putanje plovila pomoću D* lite algoritma.

D* Lite algoritam planiranja putanje se ne temelji na izvornom D*, ali implementira isto ponašanje te je danas najčešće u uporabi svih algoritama planiranja putanje zasnovanih na mrežama (eng. grid-based) jer objedinjuje mnoge dobre karakteristike njegovih prethodnika. Jednostavan je za razumijevanje i može se implementirati u manje redaka koda od ostalih algoritama, otuda i naziv "D* Lite". D* Lite je inkrementalni heuristički algoritam pretraživanja u čijoj je osnovi lifelong Planning A* (LPA*) algoritam planiranja putanje. Inkrementalne heurističke metode pretraživanja koriste heuristiku za fokusiranje pretraživanja i ponovnu upotrebu informacija iz prethodnih pretraživanja što je puno brži proces nego li rješavanje svakog zadatka pretraživanja ispočetka. LPA* pretražuje prostor u smjeru od polazišne do odredišne točke i stoga su njegove g-vrijednosti procjene početnih udaljenosti. Za razliku od LPA*, D* Lite pretražuje prostor od odredišne do polazišne točke i stoga su njegove g-vrijednosti procjene ciljnih udaljenosti. Postoji nekoliko stavki koje ga izdvaja u odnosu na njegove prethodnike, a to su efikasnost u realnom vremenu što kazuje da može brzo reagirati na promjenu okruženja, adaptivan je i fleksibilan te je zbog svoje brzine izvođenja i mogućnosti replaniranja početne rute iznimno pouzdan i siguran.

Algoritam D* lite se u procesu testiranja algoritama globalnog planiranja putanje pokazao kao vremenski optimalan što je veoma bitna stavka koja je presudila njegovom odabiru. Naime, područja pretrage algoritma globalnog planiranja putanje plovila višestruko su veća od područja pretrage mobilnih roboti ili autonomnih automobila. Mobilni roboti planiraju putanju na prostorima do nekoliko stotina metara duljine i širine, a autonomni automobili imaju predefinirane rute jer su uvjetovane prometnicama i prometnim pravilima. S druge strane, planiranje se globalne putanje plovila vrši u skalamama prostora od nekoliko tisuća do nekoliko stotina tisuća kilometara. Upravo je zato vremenska optimizacija veoma bitan čimbenik cjelokupnog procesa jer je planiranje procesno i vremenski zahtjevno te su razlike u vremenu računanja algoritama na većim skalamama prostora izražene. Također, pravilne prilagodbe algoritma mogu skratiti vrijeme računanja od nekoliko sekundi do nekoliko minuta, što je važna stavka kod korisničkog iskustva. Drugi je razlog zašto je odabran D* lite je prethodno spomenuta mogućnost preplaniranja putanje. Iako je u ovom radu obrađena tema globalnog planiranja putanje, uvezši u obzir široku sliku planiranja putanje plovila koje uključuje globalno i lokalno planiranje putanje, D* lite je logički slijedan odabir. Razlog tome je što algoritam ima

mogućnost lokalnog replaniranja putanje u dinamičkim okruženjima, dok je istovremeno zasnovan na principima globalnog planiranja putanje. U ovome radu nije ispitan proces replaniranja putanje, međutim algoritam ima mogućnost ažuriranja položaja dinamičkih prepreka, što može biti izuzetno korisno u budućim testnim fazama projekta prilikom izrade algoritama lokalnog planiranja putanje. Zaključno, D* lite objedinjuje dvije veoma bitne karakteristike koje su presudile njihovom odabiru kao glavnog algoritma planiranja putanje ovog rada, a to su brzina izvođenja i mogućnost prilagodbe u narednim fazama projekta za lokalno replaniranje putanje. U nastavku će biti opisana tehnička strana algoritma te prilagodba na mapu cijena prostora.

Osnovni D* lite algoritam, kao i algoritmi u poglavlju 2.5.2. preuzet je s GitHub repozitorija PythonRobotics koji je opisan u radu [2]. Navedeni je algoritam izrađen po principima kojeg su 2002. godine predstavili Sven Koenig i Maxim Likhachev u njihovom radu pod naslovom "D* Lite," koji je objavljen na Nacionalnoj konferenciji o umjetnoj inteligenciji (AAAI) 2002. godine. Detaljniji opis funkcionalnosti D* lite algoritma moguće je pronaći u radu [3]. U okviru ovog rada neće biti detaljnije opisana sama osnovna funkcionalnost D* lite algoritma jer je široko poznata i dostupna u mnogim resursima i znanstvenim radovima. Međutim, bit će opisane prilagodbe algoritma za rad na mapi cijena te kako one utječu na pojedine funkcije i parametre osnovnog D* lite algoritma.

D* lite algoritam iz rada [2] je izrađen u demonstracijske svrhe za primjenu na binarnoj mapi manjeg prostora te su stoga bile potrebne određene prilagodbe kako bi algoritam funkcionirao na mapi cijena većih skala. Algoritam je kroz pripadajuće funkcije i globalne varijable integriran u serverski čvor *path_planning_client.py* kako bi se mogle razmjenjivati povratne informacije i rezultati s akcijskim klijentom. Nakon procesa inicijalizacije, dostupne su informacije u globalnim varijablama o mapi cijena prostora te polazišnoj i odredišnoj točki u lokalnom koordinatnom sustavu. Lokalni se koordinati sustav koristi kako bi se izradila mreža čvorova (eng. nodes) koja predstavlja prostor u kojem se planiranja putanja plovila. Čvorovi su osnovni elementi mreže i predstavljaju diskretne točke u prostoru te su prikazani u obliku zasebne klase iz *Node.py* datoteke. Navedena klasa u konstruktoru inicijalizira koordinate čvora i cijenu, a također posjeduje funkcije zbrajanja i usporedbe koordinata čvorova. Zbog navedenog, prvi čvorovi mreže koji se postavljaju jesu polazišna i odredišna točka putanje.

Nadalje, bitne stavke algoritma su troškovi čvorova jer se pomoću njih mapa cijena integrira u algoritam. Postoji nekoliko vrsta troškova, od kojih je prvi stvarni **g-trošak** koji naznačuje trošak putanje od početnog čvora planiranja putanje do trenutnog čvora. Početni čvor planiranja putanje može zbuniti jer u stvarnosti on predstavlja odredišnu točku putanje jer algoritam istražuje prostor od odredišne do polazišne točke putanje. Za razumijevanje koncepta prilagodbe putanje navedeno je potrebno imati na umu. Ukupni se g-trošak prilikom računanja povezuje s više manjih *c-troškova* koji naznačuju trošak prijelaza između dva čvora. Mapa cijena je pri tome bitan čimbenik određivanja c-troška jer ona definira cijenu narednog čvora. Prvi je korak definiranje faktora m i n na temelju pozicije čvora (x_2, y_2) , u ovisnosti je li navedeni čvor obale, crvene, žute, zelene i sigurne zone ili ništa od navedenog. Vrijednosti se faktora definiraju s obzirom na pretvodno postavljeni rječnik cijena, odnosno ako je koordinate čvora moguće dohvatiti u rječniku inicijaliziraju se vrijednosti m i n kao cijena i korak prolaska iz rječnika. Ukoliko čvor nije moguće pronaći u rječniku smatra se da koordinate pripadaju otvorenome moru u kojemu nema ograničenja te se zato vrijednost cijene postavlja na 1 (najmanja moguća) te korak kao maksimalno definirani. Proces dohvata podataka iz rječnika cijena je opisan u obliku jednadžbe 2.11 :

$$m, n = \text{factor} \quad \text{gdje} \quad \text{factor} = \text{cost_dictionary}((x_2, y_2), [1, \text{open_sea_step}]) \quad (2.11)$$

Nakon sredjivanja vrijednosti m i n faktora, izračunava se novi čvor koji kao vrijednost koordinata kombinira faktor koraka n i razliku koordinata trenutnog i narednog čvora čime se efektivno koordinata novog čvora translatira u ishodište koordinatnog sustava. Razlog translacije koordinate novog čvora u ishodište je usporedba sa smjerovima kretanja. Jednadžba je prikazana u obliku 2.12 :

$$\text{new_node} = (n \cdot (x_1 - x_2), n \cdot (y_1 - y_2)) \quad (2.12)$$

Lista kretanja *motions* sadrži sve moguće smjerove kretanja (motions) od trenutne pozicije čvora, gdje svaki pomak uključuje novu poziciju (koordinate) i pripadajući tro-

šak u ovisnosti o postavljenim faktorima m i n . Lista mogućih pomaka s trenutne pozicije čvora definirana je kao:

$$\text{motions} = \left\{ \begin{array}{l} \text{Node}((n, 0), m) \\ \text{Node}((0, n), m) \\ \text{Node}((-n, 0), m) \\ \text{Node}((0, -n), m) \\ \text{Node}((n, n), m\sqrt{2}) \\ \text{Node}((-n, n), m\sqrt{2}) \\ \text{Node}((n, -n), m\sqrt{2}) \\ \text{Node}((-n, -n), m\sqrt{2}) \end{array} \right.$$

Gdje su:

- $\text{Node}((n, 0), m)$: Pomak udesno za n jedinica, trošak je m .
- $\text{Node}((0, n), m)$: Pomak gore za n jedinica, trošak je m .
- $\text{Node}((-n, 0), m)$: Pomak ulijevo za n jedinica, trošak je m .
- $\text{Node}((0, -n), m)$: Pomak dolje za n jedinica, trošak je m .
- $\text{Node}((n, n), m\sqrt{2})$: Pomak dijagonalno gore-desno za n jedinica, trošak je $m\sqrt{2}$.
- $\text{Node}((-n, n), m\sqrt{2})$: Pomak dijagonalno gore-lijevo za n jedinica, trošak je $m\sqrt{2}$.
- $\text{Node}((n, -n), m\sqrt{2})$: Pomak dijagonalno dolje-desno za n jedinica, trošak je $m\sqrt{2}$.
- $\text{Node}((-n, -n), m\sqrt{2})$: Pomak dijagonalno dolje-lijevo za n jedinica, trošak je $m\sqrt{2}$.

Definirane koordinate novog čvora i smjerovi kretanja liste motions imaju hvatište u ishodištu koordinatnog sustava te njihove udaljenosti od ishodišta ovise proporcionalno o koraku algoritma n pa se shodno navedenom mogu usporediti. Naime, u listi motions se detektira smjer kretanja koji odgovara koordinati navedenog novog čvora. Iz liste motions se potom dohvata vrijednost cijene smjera kretanja te se vraća pozivatelju funk-

cije kao ukupna vrijednost c-troška što je prikazano jednadžbom 2.13 Drugačije rečeno, postoje osam čvorova u osam smjerova kretnje za n udaljenih od ishodišta koordinatnog sustava. Translatacijom novog čvora u ishodište koordinatnog sustava, koordinate čvora će se zasigurno poklopiti s jednim od smjerova kretnje te će cijena navedenog čvora kretnje koja ovisi o dohvaćenoj vrijednosti m iz rječnika cijena čvora biti postavljena kao g-trošak.

$$c_cost = \{motion \mid compare_coordinates(motion, new_node)\} \quad (2.13)$$

Iz prethodno navedenog g-troška i c-troška se izrađuje ukupni trošak nazvan rhs (Right-Hand-Side). Rhs vrijednost za trenutni čvor se definira kao najbolja procjena trenutnog troška prolaska do ciljnog čvora algoritma. Za razumijevanje rhs troška potrebno je definirati prostor pretrage algoritma koji se označava slovom U i predstavlja skup prethodnika trenutnog čvora kroz koje je moguće doći iz početnog čvora do trenutnog, a trenutni čvor u skupu prethodnika naznačen je s u . Vrijednost rhs trenutnog čvora definirana je jednadžbom 2.14 Funkcija c-troška računa vrijednost troška od čvora u do čvora sprime, a funkcija g-troška predstavlja akumulirani trošak putovanja do čvora sprime. Čvor sprime je varijabilan i predstavlja jedan od čvorova u skupu susjednih čvorova od trenutnog čvora u . Nапослјетку se pomoću funkcije minimizacije odabire najmanja cijena kombinacije c-troška i g-troška te se postavlja kao ukupna rhs vrijednost čvora u , što je prikazano jednadžbom :

$$\text{self.rhs}[u.x][u.y] = \min \left([self.c(u, sprime) + self.g[sprime.x][sprime.y]]_{\text{sprime} \in \text{self.succ}(u)} \right) \quad (2.14)$$

Također, bitan trošak kojeg je potrebno spomenuti je heuristički trošak **h-trošak** koji predstavlja procjenu preostalog troška od trenutnog čvor do ciljnog čvora. Primjena h-troška omogućuje algoritmima rad u realnom vremenu na velikim prostorima pretraživanja, fokusirajući se na najvažnije dijelove prostora za pretraživanje. Heuristički trošak može biti definiran na više načina u ovisnosti o problemu, a često se koristi euklidska udaljenost, Manhattan udaljenost ili druge metrike koje reflektiraju geometrijsku

ili topološku strukturu problema. U ovome radu kao heuristička funkcija je Manhattan udaljenost jer euklidska udaljenost ponekad može generirati heuristike koje precjenjuju trošak čvorova, čineći ih neprihvatljivima što se želi izbjegći. Manhattan udaljenost koja predstavlja h-trošak između trenutnog čvora (x_2 i y_2) i ciljnog čvora (x_1 i y_1) definisana je jednadžbom 2.15:

$$\text{Manhattan}(x_1, y_1, x_2, y_2) = \max(|x_1 - x_2|, |y_1 - y_2|) \quad (2.15)$$

Navedeni g-trošak, c-trošak, rhs trošak i h-trošak služe algoritmu planiranja putanje za izradu mreže čvorova prostora (područja pretrage algoritma) koja algoritmu služi za planiranje putanje od početne do ciljne točke te za rekalkulaciju rute u dinamičkim okruženjima. Nadalje, jedan od bitnijih čimbenika procesa planiranja putanje na koje utječe postavljena mapa cijena su ključevi prioriteta čvorova koji se dobivaju pozivom funkcije *caluculate_key*. Ključevi prioriteta su vrijednosti koje se koriste u algoritmima za pretraživanje putanja kako bi se odredio redoslijed obrade čvorova. Ovi ključevi omogućuju algoritmu da dinamički prilagodava redoslijed obrade čvorova na temelju trenutnih informacija o troškovima putanja i heurističkim procjenama udaljenosti do cilja. Ključ prioriteta je predstavljen u obliku duple vrijednosti s dva polja. Prvo polje je kombinacija minimalne vrijednosti g-troška ili rhs troška, heurističkog h-troška i faktora *km* koji naznačuje prioritete čvorova (jednadžba 2.16). Drugo polje predstavlja minimalnu vrijednost g-troška ili rhs troška bez heuristike i dodatnog faktora (jednadžba 2.17). Vrijednost ključa se vraća pozivatelju funkcije za izračun ključa u obliku prikazanom u jednadžbi 2.18. Pravilno izračunavanje ključa prioriteta osigurava da se čvorovi obrađuju prema njihovom stvarnom i privremenom trošku, uzimajući u obzir i heurističku procjenu udaljenosti do cilja, što omogućuje efikasno traženje najkraće putanje u promjenjivim uvjetima okoline.

$$\text{value_1} = \min(\text{self.g}[s.x][s.y], \text{self.rhs}[s.x][s.y]) + \text{self.h}(s) + \text{self.km} \quad (2.16)$$

$$\text{value_2} = \min(\text{self.g}[s.x][s.y], \text{self.rhs}[s.x][s.y]) \quad (2.17)$$

$$\text{key} = (\text{value_1}, \text{value_2}) \quad (2.18)$$

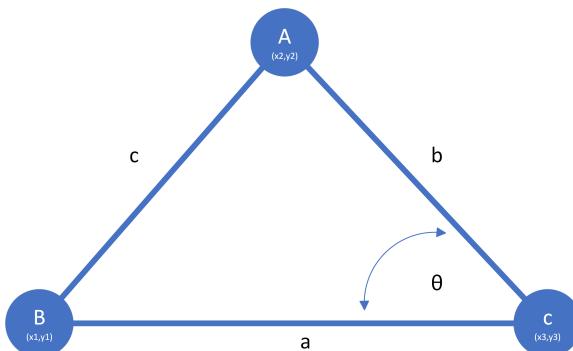
Nakon što su definirani troškovi i ključevi algoritma planiranja putanje na koje utječu postavke mape cijena, jedino što je preostalo definirati u procesu prilagodbe je područje pretrage algoritma. Područje pretrage algoritma U predstavlja listu čvorova s pripadajućim ključevima prioriteta obrade čvorova koje je do određenog trenutka algoritam planiranja putanje istražio. Ključevi prioriteta su veoma bitna stavka jer se s obzirom na njihove vrijednosti navedena lista sortira prema vrijednostima. Vizualizacija područja pretrage algoritma važan je čimbenik procesa prilikom ispravljanja pogrešaka i praćenja rada algoritma. Shodno navedenom, pretraženo se područje U putem action client i action server mehanizma komunikacije u povratnoj vraća akcijskom klijentu kao listu, ako je postavljen parametar *show_feedback* u launch datoteci.

Zaključno, u ovome su poglavlju opisane prilagodbe D* lite algoritma za rad na mapi cijena prostora te kako cijene utječu na proces računanja g-trošak, c-trošak i rhs troška. Nadalje, opisan je izabrani h-trošak u obliku Manhattan funkcije te područje pretrage U koje se kroz povratnu petlju vraća akcijskom serveru. Mapa cijena prostora dakle utječe na izgled putanje u ovisnosti o postavljenim vrijednostima cijena, što je opisano u poglavljiju 2.6.5. Međutim, kako je već prethodno naglašeno navedena putanja dobivena pomoću D* lite algoritma nije glatka te posljedično fizički izvediva jer mapa cijena ne uključuje dinamički model plovila. Kako bi se putanja izgladila te postala fizički izvediva, uведен je postupak interpolacije putanje postupcima raspoređivanja točaka i prilagođavanja krvulja koji su opisani u narednom poglavlju.

2.5.4. Interpolacija putanje postupcima raspoređivanja točaka i prilagođavanja krivulja

Interpolacija putanje plovila je proces raspoređivanja točaka putanje koju je generirao D* lite algoritam te potom prilagođavanja krivulje raspoređenim točkama kako bi se dobila glatka, fizički izvediva putanja plovila. Nakon završetka računanja putanje D* lite algoritma, serverski čvor poziva klasu *path_optimization.py* iz programske mape *curve_generation*. Klasi se prosljeđuju koordinate putanje u lokalnom koordinatnom sustavu te parametri iz launch datoteke *sampling_rate* koji definiraju koliko će točaka ne-interpolirane putanje algoritam raspoređivanja točaka te *optimization_method* pomoći kojeg se odabire jedna od šest dostupnih metoda zaglađenja krivulje.

Postupak raspoređivanja točaka temelji se na jednostavnim trigonometrijskim funkcijama te pretpostavkama glatke putanje. Trigonometrijske funkcije koje se koriste jesu euklidska udaljenost između dvije točke u prostoru te kosinusni poučak. Pomoću euklidiske udaljenosti izračunava se udaljenost između triju točaka putanje kako bi se konstruirao trokut što je prikazano na slici 2.44. gdje točka A predstavlja prvu točku putanje, C drugu, a B treću gledajući od polazišne točke putanje. Druga točka je naznačena slovom C jer je ona predmet proučavanja procesa raspoređivanja točaka, odnosno s obzirom na kut θ se izvršava pomicanje točke C ili se detektiraju promijene smjera krivulje. S druge strane, A i B u koraku izračuna nisu predmet promatranja već služe za konstrukciju navedenog trokuta. Duljina a predstavlja euklidsku udaljenost od točke B do točke C, duljina b od točke A do točke C, a duljina c od točke A do točke B te su jednadžbe proračuna duljina prikazane jednadžbama 2.19, 2.20 i 2.21. Pomoću navedenih duljina dobiva se kut θ pri vrhu C s pomoću inverznog kosinusnog poučka prikazanog jednadžbom 2.22.



Slika 2.44. Grafički prikaz trokuta za određivanje kuta pomoću kosinusnog poučka

$$a = \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2} \quad (2.19)$$

$$b = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \quad (2.20)$$

$$c = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (2.21)$$

Kut θ kod vrha C može se izračunati koristeći kosinusni poučak:

$$\theta = \arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right) \quad (2.22)$$

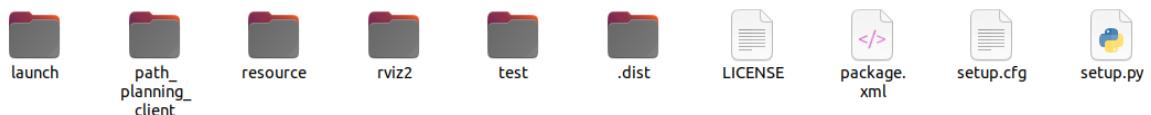
Prvo što je potrebno definirati je koje točke D* lite algoritma su problematične za izvedbu putanje plovila. Točke putanje koje predstavljaju problem jesu točke u kojima putanja naglo mijenja smjer što predstavlja problem tijekom plovid

2.6. Postavljanje polazišne i odredišne točke planiranja putanje plovila i vizualizacija podataka

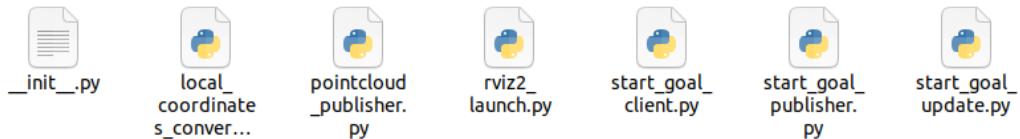
Planiranje se putanje plovila odnosi na proces određivanja optimalne rute pomoću različitih tehnika i algoritama od točke polazišta do točke odredišta. U samoj definiciji planiranja putanje plovila ističe se nekoliko pojmove. Prvi pojmovi su tehnike i algoritmi pod što spadaju algoritmi za planiranje putanje te tehnike interpolacije dobivene putanje što je opisano u poglavlju 2.5. Međutim, prije nego li se pristupi procesu planiranja putanje plovila, kao što sama definicija kazuje, potrebno je postaviti polazišnu i odredišnu točku putanje. U ovome će poglavlju upravo biti opisane tehnike postavljanja polazišne i odredišne točke planiranja putanje plovila. Između ostalog, bit će opisane tehnike vizualizacije navedenih točaka i isplanirane putanje na dobivenoj mapi geografskog prostora iz prethodnoga poglavlja 2.3. unutar RViz2 vizualizacijskog alata.

2.6.1. Struktura i pokretanje ROS2 paketa path_planning_client

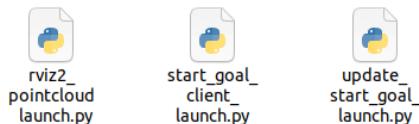
ROS2 paket *path_planning_client* zadužen je za vizualizaciju podataka unutar RViz2 vizualizacijskog alata, upravljanje procesom postavljanja polazišne i odredišne točke te objavljivanje istih unutar ROS2 okvira putem action client i action server mehanizma komunikacije za planiranje putanje plovila. Struktura paketa te pripadajućih programskih mapa launch ipath_planning_client prikazani su u nastavku na slikama 2.45., 2.46. i 2.47.



Slika 2.45. Prikaz strukture ROS2 paketa path_planning_client



Slika 2.46. Prikaz računalne mape path_planning_client



Slika 2.47. Prikaz računalne mape launch

Vizualizacija geografskih koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, polazišne i krajnje točke putanje, interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje vrši se pomoću čvora *pointcloud_publisher.py*. Detaljniji opis i tehničku izvedbu postupka vizualizacije moguće je pronaći u poglavljima 2.6.2., 2.6.3. i 2.6.4. Čvor inicijalizira preplatnike te pomoću preplatnika *start_goal_publisher* dohvaca globalne koordinate geografskog prostora objavljene na temi *gps_coordinates_coast* putem izdavača *publish_map_launch.py* iz *map_maker* paketa. Shodno navedenom, preduvjet dohvata podataka je pokretanje čvora *publish_map_launch.py*. Pomoću dohvaćenih geografskih koordinata globalnog koordinatnog sustava tvori se lokalni koordinatni sustav u klasi *local_coordinates_converter.py*.

po principu opisanom u poglavlju 2.2. te se potom globalne koordinate pretvaraju u koordinate lokalnog koordinatnog sustava. Međutim, koordinate lokalnog koordinatnog sustava nisu namijenjene prikazu u RViz2 alatu (jer su spremljene u obliku lista) već su među korak pretvorbe koordinata globalnog koordinatnog sustava u

sensor_msgs.PointCloud2 ili *geometry_msgs.PoseStamped* poruku. Kako je ranije navedeno, lokalne koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje se pretvaraju u poruku *sensor_msgs.PointCloud2*, a polazišne i krajnje točke putanje u poruku *geometry_msgs.PoseStamped*. Nadalje, pretvorene koordinate se pomoću izdavača objavljaju na pripadajućim temama namijenjenim za vizualizaciju u Rviz2 alatu.

Čvor *poincloud_publisher.py* pokreće launch datoteka *rviz2_pointcloud_launch.py*. Osim čvora pretvorbe podataka za vizualizaciju, launch datoteka poziva čvor *rviz.launch.py* koji pokreće RViz2 vizualizacijski alat sa spremlijenim postavkama vizualizacije te čvor *start_goal_publisher.py* koji je zadužen za upravljanje procesom postavljanja polazišne i odredišne točke putanje. Nakon inicijalizacije pretplatnika, čvor *start_goal_publisher.py* se pretplaćuje na teme *start_goal_msg_update* i *start_goal_msg_update_manual* te s obzirom na aktualnost objavljenih podataka objavljuje polazišnu i odredišnu točku putanje na temu *start_goal_msg*. Pojam aktualnosti se odnosi na vrijednost točaka te vremenski okvir njihovog postavljanja što je objašnjeno u poglavlju 2.6.3. Temu *start_goal_msg_update* objavljuje čvor *rviz2_pointcloud_launch.py* te sadrži podatke i vremenski okvir postavljenih točaka putanje unutar RViz2 alata. Nadalje, temu *start_goal_msg_update_manual* objavljuje čvor *start_goal_update.py* koji pokreće launch datoteka *update_start_goal_launch.py*. Pomoću launch datoteke se navedenom čvoru prosljeđuju podaci o točkama iz Linux terminala te zato tema ima sufiks *manual* kako bi se naznačilo da se podatke korisnik "ručno" objavljuje pokretanjem datoteke. Postoje mogućnosti ažuriranja samo polazišne ili odredišne točke putanje ili obje istovremeno, a čvor je moguće pokrenuti više puta sve dok korisnik nije zadovoljan postavljenim točkama. Zaključno, čvor *start_goal_publisher.py* prima ažuriranja polazišne i odredišne točke iz RViz2 alata i Linux terminala te se aktualna ažuriranja točaka postavljaju kao globalna polazišna i odredišna točka. Drugačije rečeno, moguće je primjerice postaviti polazišnu točku unutar RViz2 alata i odredišnu točku pomoću *start_goal_update.py* čvora te će navedene točke biti postavljene kao globalna polazišna i odredišna točka, iako

su postavljene iz različitih izvora.

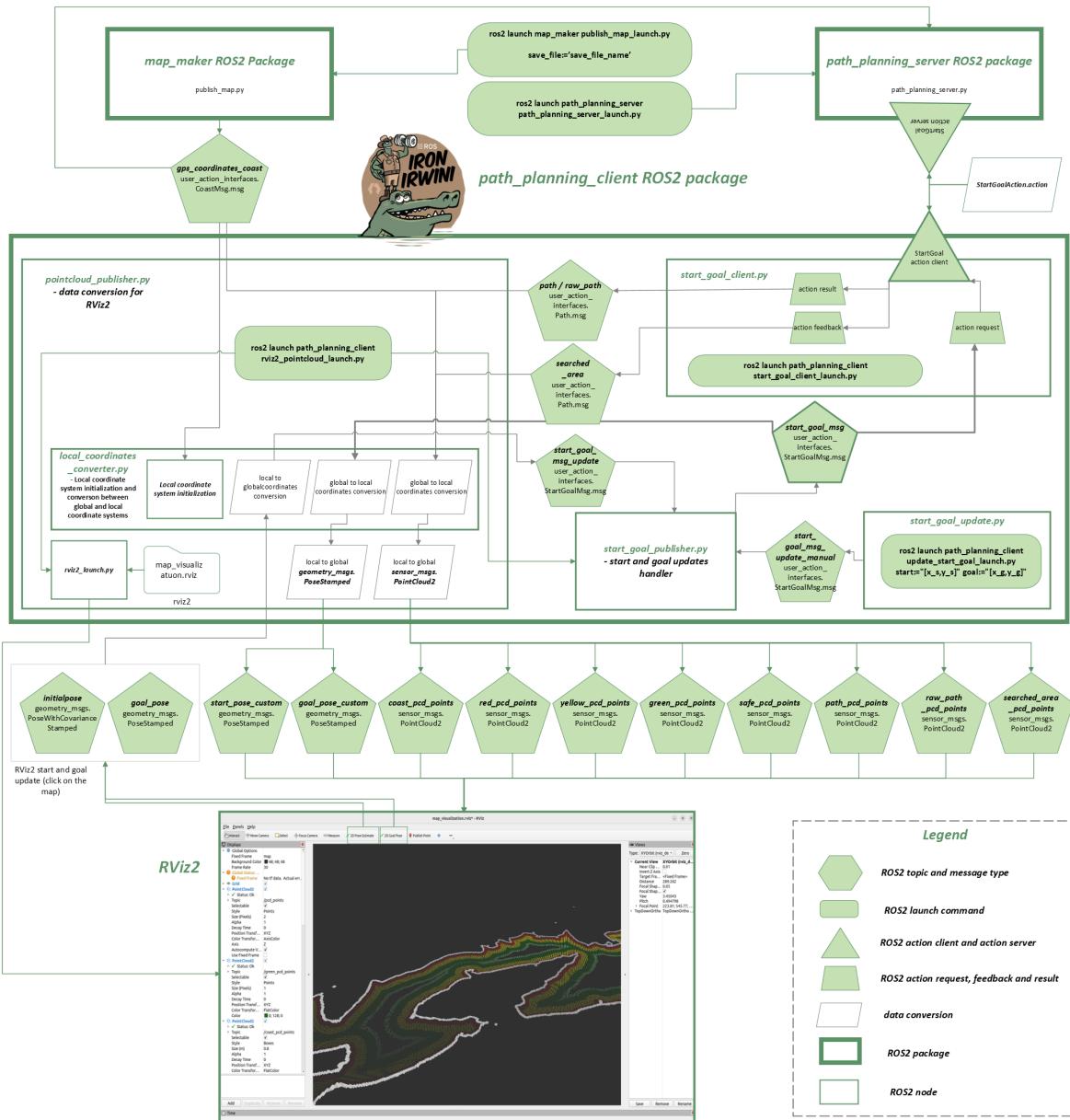
Postavljanje polazišne i odredišne točke se vrši u svrhu globalnog planiranja putanje plovila kojim presjeda action client i action server ROS2 mehanizam komunikacije. U trenutku nakon što je korisnik zadovoljan postavljenim globalnim točkama *start_goal_msg*, pokreće se launch datoteka *start_goal_client_launch.py* koji pokreće *start_goal_client.py* klijentski čvor. Navedeni čvor inicijalizira polazišnu i odredišnu točku putanje dostupne na temi *start_goal_msg* u vremenskom okviru pokretanja datoteke te akciju za planiranje putanje *start_goal_action*. U polje zahtjev inicijalizirane akcije *StartGoalAction.action* postavljaju se potom točke putanje te se pokreće asinkroni mehanizmi upravljanja komunikacijom. Mehanizam potom čeka pokretanje akcijskog servera *path_planning_server.py* iz *path_planning_server* paketa. Uspostavom komunikacije, server će kao ulazne podatke inicijalizirati točke iz zahtjeva te u povratnoj petlji vraćati područje pretrage algoritma planiranja putanje i naposljetku vratiti rezultat dobivene putanje u interpoliranom i neinterpoliranom obliku. Detaljniji opis komunikacije akcijskog klijenta i servera su opisani u poglavljima 2.6.3. i 2.6.4. Grafički prikaz arhitekture komunikacije akcijskog čvorova, mehanizma akcijskog klijenta i server te obrade i pretvorbe podataka uz navedene naredbe za pokretanje paketa *path_planning_client* prikazan je na slici 2.48.

Popis ROS2 naredbi za pokretanje ROS2 paketa path_planning_clienta

Za pokretanje RViz2 vizualizacijskog alata koristi se *rviz2_pointcloud_launch.py* launch datoteka. Launch datoteka poziva pripremljenu RViz2 datoteku *map_visualization.rviz* iz *rviz2* programske mape sa spremnjim postavkama za vizualizaciju zadanih tema te ranije navedeni čvor za pretvorbu podataka *pointcloud_publisher.py*. Launch datoteka se pokreće naredbom:

```
ros2 launch path_planning_client rviz2_pointcloud_launch.py
```

Uz navedenu naredbu potrebno je pokrenuti i objavljivanje mape geografskog prostora iz **map_maker** paketa što je osnova stvaranja lokalnog koordinatnog sustava:



Slika 2.48. Arhitektura komunikacije ROS2 paketa `path_planning_client`

```
ros2 launch map_maker publish_map.launch.py
save_file:='save_file_name'
```

Za pokretanje ažuriranja polazišne i odredišne točke putanje pomoću Linux terminala pokreće se naredba gdje x_s , y_s , x_g i y_g predstavljaju geografske koordinate polazišne i odredišne točke putanje :

```
ros2 launch path_planning_client update_start_goal_launch.py  
start:="[x_s,y_s]"
```

```
ros2 launch path_planning_client update_start_goal_launch.py  
goal:="[x_g,y_g]"
```

```
ros2 launch path_planning_client update_start_goal_launch.py  
start:"[x_s,y_s]" goal:="[x_g,y_g]"
```

Ažuriranje točaka moguće je i direktno putem RViz2 alata klikom na polja alatne trake *2D Pose Estimation* i *2D goal pose* te postavljanjem oznaka na željene lokacije prikazane mape geografskog prostora.

Naposljetku, ukoliko je korisnik zadovoljan postavljenom polazišnom i odredišnom točkom, pokreće se sljedeća naredba, čija je svrha uspostava komunikacije putem akcijskog klijenta s akcijskim servom zaduženim za planiranje putanje plovila:

```
ros2 launch path_planning_client start_goal_client_launch.py
```

2.6.2. Vizualizacija podataka u RViz2 vizualizacijskom alatu

Shodno ranije navedenoj definiciji uloge *path_planning_client* paketa iz prethodnog poglavlja 2.45., prva je zadaća paketa vizualizacija željene mape geografskih koordinata, polazišne i odredišne točke (koordinate) te isplanirane putanje plovila. Vizualizaciju pokreće launch datoteka *rviz2_poincloud.launch.py* koja poziva rviz datoteku *map_visualization.rviz* (iz *rviz2* programske mape) sa spremnjim postavkama za vizualizaciju geografskih koordinata obale, zona udaljenosti od obale, polazišne i odredišne točke putanje, područja pretrage algoritma planiranja putanje te isplanirane putanje u interpoliranom i neinterpoliranom obliku. Nadalje, poziva se čvor *pointcloud_publisher.py* koji vrši prilagodbu i objavljivanje podataka pogodnih za vizualizaciju u RViz2 alatu te čvor *start_goal_publisher* koji je zadužen za upravljanje postavlja-

nja polazišne i odredišne točke putanje, o čemu će biti više rečeno u poglavlju 2.6.3. Kako bi se podaci uistinu mogli vizualizirati, potrebno ih je pretvoriti u oblik pogodan za vizualizaciju za što je zadužen čvor *pointcloud_publisher.py*. Pokretanjem launch datoteke, u konstruktoru čvora se inicijaliziraju preplatnici na teme čije će poruke biti dohvaćene te potom pretvorene u druge vrste poruka pogodnih za vizualizaciju. Naziv preplaćenih tema s pripadajućim vrstama poruke i opisom podataka prikazani su u tablici 2.9.

Naziv teme	Vrsta poruke	Opis podataka
'gps_coordinates_coast'	CoastMsg	Koordinate obale i zona udaljenosti
'start_goal_msg'	StartGoalMsg	Geografske koordinate polazišne i odredišne točke
'path'	PathMsg	Geografske koordinate interpolirne putanje
'raw_path'	PathMsg	Geografske koordinate neinterpoliranog algoritma
'searched_area'	PathMsg	Podaci o pretraženom području algoritma za planiranje putanje
'initialpose'	PoseWithCovarianceStamped	Postavljena polazišna točka u RViz2
'goal_pose'	PoseStamped	Postavljena odredišna točka u RViz2

Tablica 2.9. Tablica preplaćenih poruka sa nazivima tema, vrstama poruka i opisima

Iz stupca *Vrste poruka* tablice 2.9. je vidljivo da osim poruke CoastMsg.msg čija je struktura definirana u poglavlju 2.3.5. postoje vrste poruka koje je tek potrebno definirati, poput CoastMsg.msg, poruke StartGoalMsg.msg i PathMsg.msg također su dijelom *user_action_interfaces* ROS2 paketa. Nadalje, poruke PoseWithCovarianceStamped i PoseStamped su standardne poruke ROS2 okvira te se njihov sadržaj može pronaći na ROS2 službenim internetskim stranicama.

Poruka StartGoalMsg.msg služi za prikaz ažuriranih geografskih koordinata polazišne i odredišne točke putanje te globalne polazišne i odredišne točke putanje plovila. Sadržaj globalne polazišne i odredišne točke putanje akcijski klijent predaje akcijskom serveru u svrhu planiranja putanje plovila što je opisano u poglavlju 2.6.3. Sadržaj poruke prikazan je u tablici 2.10.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32[]	start
float32[]	goal

Tablica 2.10. Struktura ROS poruke *StartGoalMsg.msg*

Poruka *PathMsg.msg* se koristi za prikazivanje dobivene putanje plovila u interpoliranom i neinterpoliranom obliku koje u polju rezultat vraća akcijski server akcijskom klijentu. Osim za objavu rezultata planirane putanje, poruka se koristi i za prikazivanje područja pretrage algoritma planiranje putanje koje akcijski server također kao povratnu informaciju vraća akcijskom klijentu. Sadržaj poruke prikazan je u tablici 2.11.

Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32[]	path_x
float32[]	path_y

Tablica 2.11. Struktura ROS2 poruke *PathMsg.msg*

Inicijalizacijom ranije opisanih pretplatnika, ujedno se inicijaliziraju i njihovi pozivi. Nakon pokretanja čvora u odgovarajućim vremenskim trenutcima pozivi dohvaćaju poruke ROS2, obrađuju ih i spremaju u globalne varijable čvora. Obrada poruka podrazumejava raspakiravanje podataka i spremanje u pogodan oblik za obradu.

Osim inicijaliziranih pretplatnika, inicijaliziraju se globalne varijable i izdavači, čija je uloga objavljivanje poruka pretvorenih podataka pogodnih za vizualizaciju te slanje poruka o ažuriranim polazišnim i odredišnim koordinatama u RViz2 alatu. Nazivi tema za objavljivanje s pripadajućim vrstama poruka pogodnih za vizualizaciju podataka prikazani su u tablici 2.12.

Kako bi proces pretvorbe i vizualizacije uopće i otpočeo potrebno je pokrenuti objavljivanje podataka geografskog prostora iz *map_maker* paketa postupkom opisanim u 2.3.5. Nakon objavljivanja podataka mape geografskog prostora, započinje pretplata na temu *gps_coordinates_coast* te poziv pretplatnika samo jednom dohvaća poruka i spremi podatke o geografskim koordinatama obale i zona udaljenosti u globalne varijable čvora.

Naziv teme	Vrsta poruke
'coast_pcd_points'	sensor_msgs.PointCloud2
'red_pcd_points'	sensor_msgs.PointCloud2
'yellow_pcd_points'	sensor_msgs.PointCloud2
'green_pcd_points'	sensor_msgs.PointCloud2
'safe_zone_pcd_points'	sensor_msgs.PointCloud2
'path_pcd_points'	sensor_msgs.PointCloud2
'raw_path_pcd_points'	sensor_msgs.PointCloud2
'searched_area_pcd_points'	sensor_msgs.PointCloud2
'start_pose_custom'	PoseStamped
'goal_pose_custom'	PoseStamped
'start_goal_msg_update'	StartGoalMsg

Tablica 2.12. Tablica poruka za objavljivanje sa nazivima tema i vrstama poruka

Ovakav pristup dohvaćanja poruke usvojen je kako bi mapa geografskog prostora bila dostupna vizualizacijskom alatu i u slučaju prekida objavljivanja podataka na temu *gps_coordinates_coast*. S druge strane, poziv ima mogućnost prepoznavanja novih podataka geografskog prostora te se u tom slučaju ponovo inicijaliziraju globalne varijable čvora kako bi se tijekom procesa mogle vizualizirati različite mape geografskog prostora bez ponovnog pokretanja *rviz2_poincloud_launch.py* launch datoteke.

Dohvaćanjem poruke s podacima o geografskom prostoru započinje proces pretvorbe podataka. Prvi je korak procesa spajanje razdvojenih lista geografskih koordinata obale i zona udaljenosti u višedimenzionalna polja. Na osnovi ovog koraka stvara se lokalni koordinatni sustav za što je zadužena klasa *local_coordinates_converter.py* po principima opisnim u poglavlju 2.2. klasa *local_coordinates_converter.py* inicijalizira liste obale i zona udaljenosti, spaja ih u jednu listu te postavlja rubne koordinate kao minimalne i maksimalne vrijednosti spojene liste. Nadalje, iz rubnih koordinata se stvara lokalni koordinatni sustav te se pomoću funkcija pretvorbe koordinate globalnog koordinatnog sustava pretvaraju u koordinate lokalnog koordinatnog sustava.

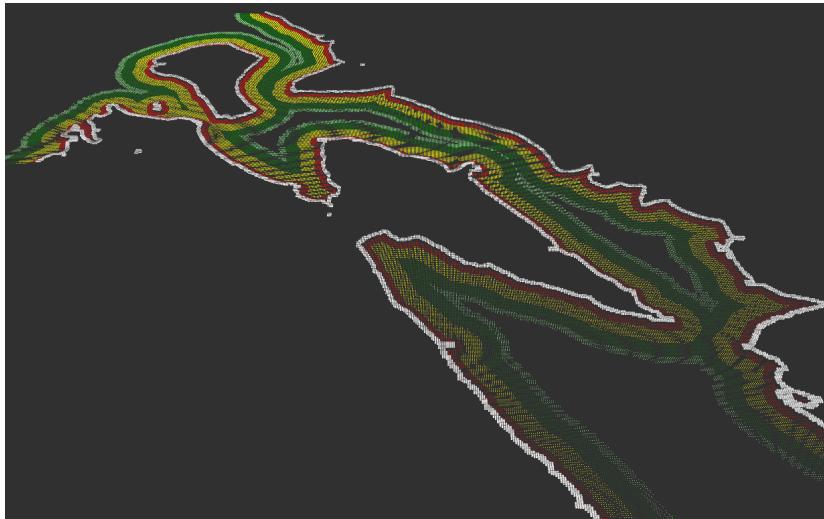
Liste koordinata obale i zona udaljenosti lokalnog koordinatnog nisu pogodne za vizualizaciju u RViz2 alatu te se zato pretvaraju u oblik poruke *sensor_msgs.PointCloud2*. Poruka *sensor_msgs.PointCloud2* je standardna poruka koja se koristi za predstavljanje skupa točaka u trodimenzionalnom prostoru prostoru. Obično se koristi za predstavljanje podataka iz senzora kao što su LiDAR-i ili 3D kamere te predstavljanje geografskih koordinata. Svaka točka u oblaku može sadržavati više polja. Polje

Fields definira tipove svakog polja u oblaku točaka (Uobičajena polja uključuju 'x', 'y', 'z' i opcionalno 'rgb' i 'intensity'), a polje bigendian koje označava je li format podataka big-endian ili little-endian. Nadalje, polje Point_step naznačuje je li veličina jedne točke u bajtovima, dok Row_step označava veličinu jednog reda u bajtovima. Polje Data sadrži stvarne podatke oblaka točaka serijalizirane u nizu bajtova. Polje Is_dense označava postoje li nevaljane točke u podatcima. Detaljniji se opis sadržaja poruke može pronaći na ROS2 internetskim stranicama.

Proces pretvorbe koordinata lokalnog koordinatnog sustava u poruku `sensor_msgs.PointCloud2` vrši funkcija `convert_to_pcd`. U funkciji je pretvorbe potrebno koordinatama postaviti treću dimenziju `height`. Vrijednost se treće dimenzije postavlja u ovisnosti o preferencijama trodimenzionalnog prikaza koordinata. Koordinate se potom pretvaraju u polje vrste NumPy array te se serijalizira u niz bajtova. Nadalje, inicijalizira se polje fields pomoću Python-ove liste comprehensions za kreiranje liste objekata tipa `sensor_msgs.PointField`. Svaki objekt predstavlja jedno polje u `PointCloud2` poruci, definirajući kako su pojedini atributi (x, y, z) točaka pohranjeni. Na kraju se postavljaju vrijednosti `point_step` i `row_step`, izrađuje se `sensor_msgs.PointCloud2` poruka koja se vraća pozivatelju funkcije.

Proces pretvorbe završava spremanjem poruke `sensor_msgs.PointCloud2` u globalne varijable čvora te objavom istih na temama prikazanim u tablici 2.12. Datoteka `textit-map_visualization.rviz` (iz `rviz2` programske mape) sadrži definirane postavke za vizualizaciju pretvorenih podataka pa se objavom na zadane teme podatci automatski prikazuju unutar RViz2 alata. Primjer trodimenzionalnog prikaza koordinata obale i zona udaljenosti unutar RViz2 vizualizacijskog alata nalazi se na slici 2.49.

Shodno ranije navedenom, u RViz2 vizualizacijskom se okruženju pomoću poruke `sensor_msgs.PointCloud2` vizualiziraju koordinate obale, crvena, žuta, zelena i sigurna zona udaljenosti, koordinate interpolirane i neinterpolirane putanje te područja pretrage algoritma planiranja putanje. Kako su navedeni podatci skupovi koordinata, vrijede ista pravila pretvorbe koordinata iz geografskih koordinata u koordinate lokalnog koordinatnog sustava te potom u `sensor_msgs.PointCloud2`. Jedini podatci za koje nije korišten oblak točaka za vizualizaciju jesu koordinate polazišne i odredišne točke. Proces njiho-



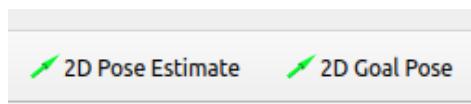
Slika 2.49. Vizualizacija obale i zona udaljenosti u RViz2 vizualizacijskom alatu
vog postavljanja i vizualizacije bit će opisan u narednom poglavlju 2.6.3.

2.6.3. Postavljanje polazišne i odredišne točke te pokretanje ROS2 action client i action server za planiranje putanje plovila

Kako i sam naslov ovog poglavlja naznačuje, glavna uloga paketa *path_planning_client* je postavljanje polazišne i odredišne točke koje će algoritam koristiti za planiranje putanje plovila. Procesom postavljanja točaka putanje upravlja čvor *start_goal_publisher.py* koji pokreće *rviz2_pointcloud_launch.py* launch datoteke zajedno s čvorom pretvorbe podataka za vizualizaciju *pointcloud_publisher.py* te čvorom *rviz_launch.py* koji pokreće RViz2 vizualizacijski alat sa spremlijenim postavkama vizualizacije.

Postavljanje polazišne i odredišne točke putanje je moguće iz dva različita izvora. Prvi je postavljanje točaka u Rviz2 alatu. Naime, Rviz2 alat ima dvije različite opcije u alatnoj traci za objavljivanje podataka o označenoj poziciji na geografskoj mapi. Sam alat objavljuje pozicije postavljenih točaka na temama na koje se lako preplatiti, stoga je idealan u svrhu postavljanja odredišne i polazišne točke. Za polazišnu točku odabrana je opcija na alatnoj traci pod imenom *2D pose estimation* koja objavljuje podatke o postavljenoj poziciji na mapi (postavlja se klikom miša na željenu poziciju) na temu *initialpose*. Vrsta poruke objavljene na temi je *geometry_msgs.PoseWithCovarianceStamped* koja se koristi za predstavljanje položaja (pozicije i orijentacije) robota ili bilo kojeg objekta u trodimenzionalnom prostoru, zajedno s kovarijacijskom matricom koja označava nesigurnost

procjene položaja. U narednim bi fazama projekta umjesto postavljenje pozicije u RViz2 alatu trebala biti izrađena tema na koju se objavljuju informacije sa GPS uređaja stvarnog plovila. Upravo zato je *geometry_msgs.PoseWithCovarianceStamped* korisna za predstavljanje polazišne pozicije putanje jer uključuje kovarijacijsku matricu nesigurnosti određenja pozicije što je karakteristika svakog senzora. Nadalje, prilagodba programa za pretplatu na novo izrađenu temu koja sadrži GPS podatke uređaja umjesto na temu *initialpose* relativno jednostavna jer navedene teme imaju istu vrstu poruke. Za odredišnu je točku putanje odabrana opcija na alatnoj traci pod imenom *2D Goal Pose* koja objavljuje podatke o postavljenoj poziciji na mapi (postavlja se klikom miša na željenu poziciju) na temu *goal_pose*. Vrsta poruke objavljena na temi je *geometry_msgs.PoseStamped* koja za razliku od *geometry_msgs.PoseWithCovarianceStamped* nema uključenu nesigurnost određivanja pozicije što je također željeno vladanje jer odredišna točka, zbog same prirode procesa, ne bi trebala biti postavljena s uključenom nesigurnosti. Na navedene objavljene teme s ažuriranjima polazišne i odredišne točke pretplaćuje se čvor *pointcloud_publisher.py* koji upravlja procesom vizualizacije te sadrži podatke o pozicijama unutar lokalnog koordinatnog sustava. Obrnutim postupkom pretvorbe podataka od postupka u prethodnom poglavlju 2.6.2., koordinate postavljene u RViz2 alatu iz lokalnog koordinatnog sustava se pretvaraju u geografske koordinate globalnog koordinatnog sustava te putem izdavača objavljuju na temi *start_goal_msg_update* kao *StartGoalMsg.msg* poruku (tablica 2.10.). Dio alatne trake s *2D pose estimation* i *2D goal pose* opcijama prikazan je na slici 2.50.



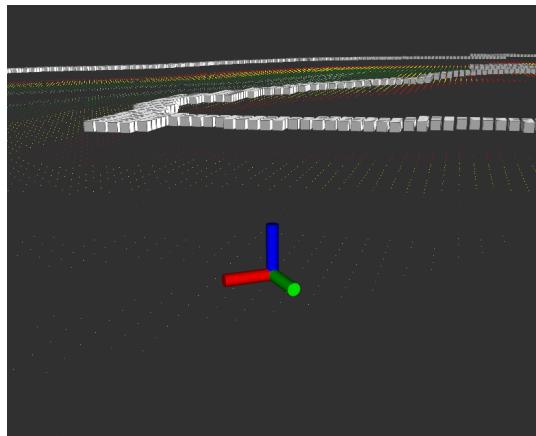
Slika 2.50. Rviz2 alatna traka

Druga je opcija postavljanja polazišne i odredišne točke putem čvora *start_goal_update* koji se pokreće putem *update_start_goal_launch.py* launch datoteke. Launch datoteka ima opciju prosljeđivanja podataka iz Linux terminal naredbe o polazišnoj točki putanje, odredišnoj točki putanje ili obje istovremeno. Navedeni program provjerava valjanost proslijedjenih podataka te ih postavlja u *StartGoalMsg.msg* poruku čiji sadržaj objavljuje na temi *start_goal_msg_update_manual*. Sufiks *manual* naznačuje

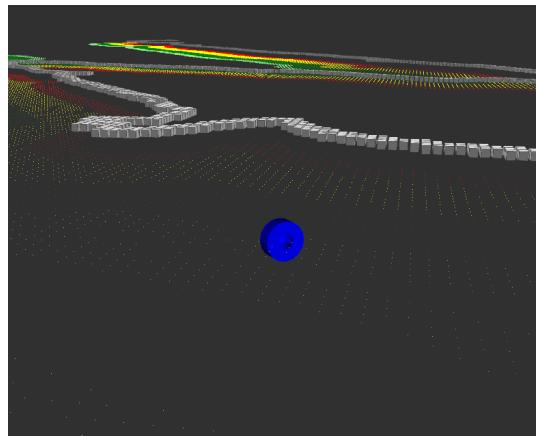
da su podaci objavljeni "ručno", tj. da ih je korisnik unio putem Linux terminal naredbe. Poruka se objavljuje samo kratak vremenski period dovoljan da *start_goal_publisher.py* dohvati objavljene podatke te ih spremi interno u globalne variabile čvora. Nakon što istekne vremenski period objavljivanja, čvor se gasi kako bi se moglo postaviti nove informacije o polazišnoj i odredišnoj točki putanje bez potrebe za prisilnim prekidanjem programa.

ROS2 čvor *start_goal_publisher.py* zadužen za postavljanje globalne odredišne i polazišne točke putanje se pretplaćuje na *start_goal_msg_update* i *start_goal_msg_update_manual* te osluškuje ažuriranja podataka na navedenim temama. Svakim ažuriranjem vrijednosti na temama, novi podatci se zajedno s vremenskim okvirom postavljaju u globalne variabile čvora. Vremenski je okvir značajan čimbenik procesa jer on određuje aktualnost podataka, odnosno koje su informacije i s koje teme posljednje ažurirane. S obzirom na aktualnost podataka, inicijalizirani je izdavač programa zadužen za objavljivanje globalne početne i polazišne točke, u pripadajućem pozivu objavljuje *StartGoalMsg.msg* poruku na temu *start_goal_msg*. Na temu je pretplaćen i *poincloud_publisher.py* koji geografske koordinate polazišne i odredišne točke globalnog koordinatnog sustava pretvara u koordinate lokalnog koordinatnog sustava te potom u *geometry_msgs.PoseStamped* poruku pogodnu za vizualizaciju u RViz2 alatu. Navedene se poruke objavljuju na teme *start_pose_custom* i *goal_pose_custom*, a potom se vizualiziraju u Rviz2 2 alatu, što je prikazano na slikama 2.55. i 2.56.

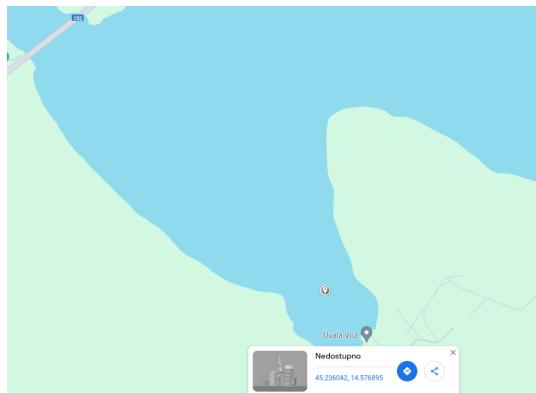
Kako bi se dokazala valjanost postavljenih geografskih koordinata putem ažuriranja u Linux terminalnu pomoću *start_goal_update.py* čvora proveden je eksperiment koji uključuje dohvaćanje geografskih podataka s javno dostupne Google Maps internetske stranice. Geografske koordinate polazišne i odredišne točke putanje preuzete su s Google Maps internetske stranice (slike 2.53. i 2.54.) te potom zalipljene u naredbu za pokretanje *update_start_goal_launch.py* launch datoteke. Ažurirane polazišne i odredišne točke u Rviz2 alatu prikazane na slikama 2.55. i 2.56. odgovarale su geografskim koordinatama iz Google Maps alata, čime je dokazano da sustav ima mogućnost upravljanja stvarnim geografskim koordinatama postavljenim iz drugih izvora podataka.



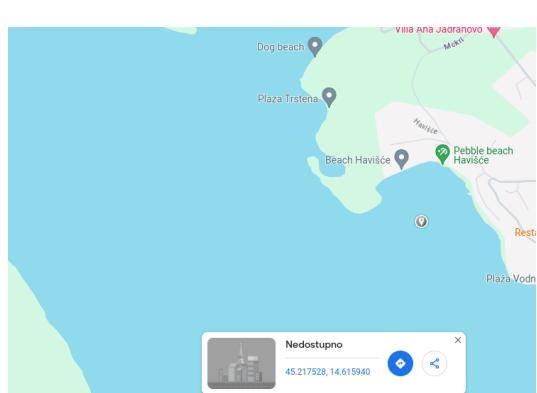
Slika 2.51. Vizualizacija polazišne točke putanje u RViz2 alatu



Slika 2.52. Vizualizacija odredišne točke putanje u RViz2 alatu



Slika 2.53. Postavljena polazišna točka u Google Maps alatu



Slika 2.54. Postavljena odredišna točka u Google Maps alatu



Slika 2.55. Vizualizacija polazišne točke iz Google Maps alata u Rviz2 alatu



Slika 2.56. Vizualizacija odredišne točke iz Google Maps alata u Rviz2 alatu

Vizualizacijom polazišne i odredišne točke u trodimenzionalnom prostoru na prikazanoj geografskoj mapi korisnik može ocijeniti valjanost postavljenih točaka pa ih navedenim postupcima ažurirati. U trenutku kada je korisnik zadovoljan postavljenim točkama pokreće se **ROS2 mehanizam komunikacije action client i action server** koji upravlja procesom planiranja putanje. Mehanizam komunikacije se pokreće putem `start_goal_client_launch.py` launch datoteke koja poziva ROS čvor `start_goal_client.py`. Čvor u konstruktoru inicijalizira preplatnika na temu `start_goal_msg` sa globalno postavljenom polazišnom i odredišnom točkom putanje. Nadalje inicijalizira izdavače za objave interpolirane putanje i neinterpolirane putanje na teme `path` i `raw_path` te izdavač za objavljivanje područja pretrage algoritma planiranja putanje na temu `searched_area`. Naposljetku se inicijalizira i akcijski klijent `start_goal_client` s akcijom naziva `StartGoalAction.action` iz `user_action_interfaces` paketa. Akcijski klijent je ROS2 entitet koji pokreće dugotrajne zadatke na serveru. Klijent šalje zahtjeve serveru, može ih otkazati i primati povratne informacije i rezultate. Akcijski server i klijent omogućuju asinkronu komunikaciju, što je korisno za dugotrajne zadatke poput planiranja putanja plovila ili bilo kojeg zadatka koji traje duže vrijeme i zahtjeva praćenje napretka. Akcijski klijent i server komuniciraju putem akcije koja ima definiranu strukturu s poljima zahtjev, rezultat i povratna informacija. Polje zahtjev popunjava akcijski klijent te se informacija prosljeđuje akcijskom serveru koji u povratnoj vezi klijentu vraća povratne informacije te naposljetku i dobiveni rezultat. Prikaz sadržaja akcije `StartGoalAction.action` dostupan je u tablici 2.13.

Nastavno na prethodno navedeno, nakon inicijalizacije preplatnika, izdavača i akcijskog klijenta čvor `start_goal_client.py` u pozivu preuzima podatke sa `start_goal_msg` i prosljeđuje ih funkciji za slanje zahtjeva akcijskom klijentu. Funkcija potom poziva inicijaliziranu akciju, postavlja polje zahtjev te osluškuje dostupnost servera. Pokretanjem se uspostavlja asinkrona komunikacija te se serveru predaje zahtjev koji sadrži polazišnu i odredišnu točku putanje uz inicijalizaciju funkcija za primanje povratnih informacija i rezultata. Server primitkom zahtjeva započinje s procesom planiranja putanje plovila te u povratnoj petlji vraća informacije u području pretrage algoritma planiranja putanje. Područje pretrage zapravo predstavlja geografski prostor koji je algoritam planiranja pu-

Vrsta Podataka	Naziv	Tip
float32[]	start	zahtjev
float32[]	goal	zahtjev
<hr/>		
float32[]	path_x	rezultat
float32[]	path_y	rezultat
float32[]	raw_path_x	rezultat
float32[]	raw_path_y	rezultat
float32	path_distance	rezultat
float32	raw_path_distance	rezultat
float32	estimated_path_time	rezultat
float32	estimated_raw_path_time	rezultat
<hr/>		
float32[]	partial_path_x	povratna informacija
float32[]	partial_path_y	povratna informacija
float32[]	search_area_x	povratna informacija
float32[]	search_area_y	povratna informacija

Tablica 2.13. Struktura ROS2 akcije *StartGoalAction.action*

tanje istražio do trenutka objavljivanja povratne informacije što je izuzetno korisno za promatranje i ispravljanje rada algoritma. Povratne se informacije spremaju u globalne varijabla čvora i putem izdavača objavljaju na temu *searched_area*. Nadalje, u trenutku nakon što je putanja isplanirana, server vraća u polju rezultat geografske koordinate interpolirane i neinterpoliran putanje uz pripadajuće informacije o duljini putanja i procijenjenom vremenu trajanja plovidbe te ih akcijski klijent sprema u globalne varijable i objavljuje na teme *path, raw_path* i *path_info* čime završava proces planiranja putanje. Putem teme *path_info* dostupne su informacije o duljini putanja i procijenjenom vremenu putem poruke *PathInfoMsg.msg* iz *user_action_interfaces* paketa čiji je sadržaj prikazan u tablici 2.14.

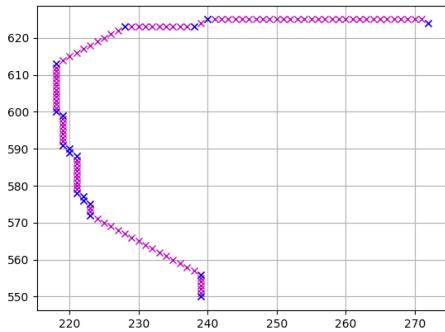
Tip podatka	Naziv
std_msgs/Header	header
string	frame_id
float32	path_distance
float32	raw_path_distance
float32	estimated_path_time
float32	estimated_raw_path_time

Tablica 2.14. Struktura ROS2 poruke *PathInfoMsg.msg*

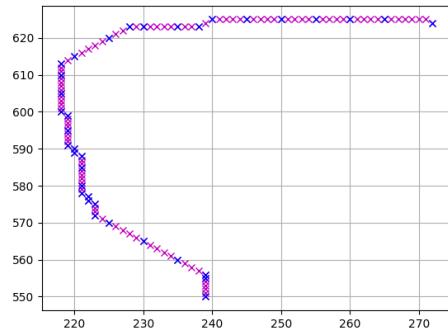
Pri tomu važan je čimbenik procesa i čvor *poincloud_publisher.py* koji pretplatom na teme iz *searched_area* omogućuje vizualizaciju područja pretrage algoritma planiranja

putanje te naposljetu i isplaniranih putanja. Područje pretrage i isplaniranih putanja se pretvaraju u lokalni koordinatni sustava te potom u poruku *sensor_msgs.PointCloud2* i putem izdavača objavljaju na temama *searched_area_pcd_points*, *raw_path_pcd_points* i *raw_path_pcd_points*. Navedene teme se vizualiziraju u Rviz2 alatu.

Zaključno, u ovome je poglavlju opisan proces vizualizacije geografskih koordinata obale, crvene, žute, zelene i sigurne zone udaljenosti, polazišne i krajnje točke putanje, interpolirane i neinterpolirane putanje kao i područja pretrage algoritma planiranja putanje. Također je objašnjen proces postavljanja polazišne i odredišne točke putanje uz action client i action server mehanizam komunikacije za planiranje putanje plovila. Jedino što je preostalo opisati u glavnom dijelu ovog rada je dio čiji naziv je ujedno i naziv rada, a to je **modelski informirano globalno planiranje putanje plovila**. Mehanizmi opisani u prethodnim poglavljima izrade mape geografskog prostora i postavljanja polazišne i odredišne točke putanje uz vizualizaciju rezultata su neizostavne sastavnice procesa planiranja putanje bez čijeg razumijevanja i implementacije sam proces nije moguć. Zbog ranije navedenog, u narednom će poglavlju biti opisan modelski informirano globalno planiranje putanje plovila u čijoj je osnovi mapa geografskih koordinata globalnog koordinatnog sustava te pripadajuća polazišna i odredišna točka putanje.



Slika 2.57. Detektirane točke promijene smjera putanje

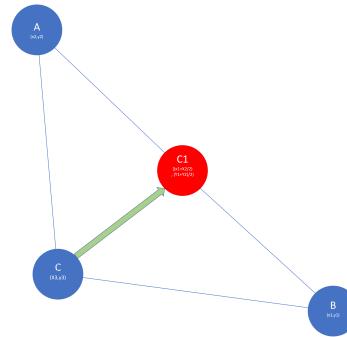


Slika 2.58. Konačni skup uzorkovanih točaka za interpolaciju putanje

be. Razlog tome je što plovila zbog dinamičkih karakteristika ne mogu trenutačno skrenuti, već je potrebno neko vrijeme za prilagodbu promjeni smjera. Drugačije rečeno, plovila imaju određeni minimalni radijus skretanja na što je potrebno obratiti pozornost. Provjerava se izvršava u iterativnom procesu gdje se za svake tri slijedne točke putanje provjerava kut θ pri drugoj točki putanje. Postupkom pokušaja i pogrešaka dobiten je zaključak da ako je kut manji od 150 stupnjeva pri određenoj točki putanje, točka zasigurno predstavlja točku promjene smjera putanje. Navedeni je postupak detekcija točaka promijene smjera prikazan slikom 2.57. gdje su ružičastom x znakovima prikazane točke putanje D* lite algoritma, a plavim detektirane točke promijene smjera. Iz slike se može zaključiti da su točke promjene smjera putanje uspješno detektirane uz navedenu premisu. Također, važan parametar prilikom uzorkovanja točaka putanje je i *sampling_rate* koji definira koliko će točaka putanje biti uzorkovano. Primjerice, ako je parametar *sampling_rate* pet, uzeti će se svaka peta točka putanje. Zaključno, konačni skup točaka za interpolaciju putanje čine točke promijene smjera koje predstavljaju problem izvedivosti putanje te točke definirane parametrom *sampling_rate* što je prikazano na slici 2.58. Utjecaj parametra *sampling_rate* na izgled putanje opisano je u poglavljju 2.6.5.

U trenutku kada je uzorkovan skup točaka D* lite algoritma započinje proces interpolacije putanje koji se sastoji od postupaka razmještanja točaka i prilagođavanja krivulje. Proces razmještanja točaka je iterativni proces, odnosno sastoji se od nekoliko programskih petlji koje provjeravaju uvjete glatkoće putanje. Prva pretpostavka fizičke izvedivosti putanje je da ne postoji nagla skretanja, odnosno putanja je glatka ako i samo ako kut

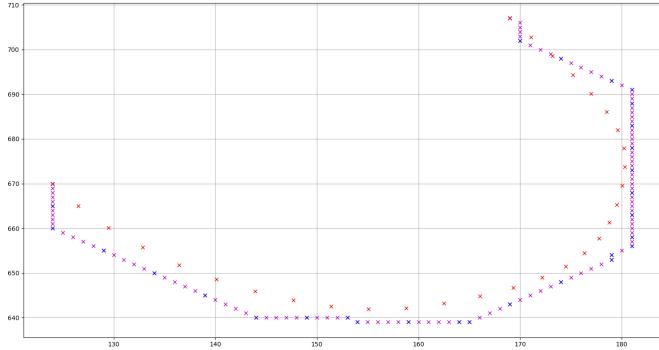
θ druge točke putanje između bilo koje tri slijedne točke putanje je veći od određene granice. Druga pretpostavka izvedivosti putanje kretanja po pravcu je da ako je kut θ tupi i veći od određene granice, točka C može biti translatirana u poziciju koja predstavlja polovičnu udaljenost između točke A i B kako bi tri navedene točke činile pravac što je prikazano grafički na slici 2.59. Druga pretpostavka glatkoće putanje lako se implementira u algoritam razmještanja točaka. Naime, postupkom pokušaja i pogrešaka dobivena je zaključak da ako je kut veći od 150 stupnjeva sigurno ne predstavlja točku promijene gradijenta tj. smjera putanje. Ako navedena točka ne predstavlja točku promijene smjera putanje posljedično znači da da navedena točka zapravo odstupa za mali pomak od pravca putanje. Shono navedenom, u iterativnom algoritmu se za svake tri slijedne točke putanje počevši od polazišne točke putanje (osim završne dvije točke putanje jer nije moguće konstruirati trokut) provjerava je li kut pri srednjoj točki C veći od 150 stupnjeva. Ukoliko to je slučaj, točka se translatira u novu koordinatu na polovini udaljenosti između prve i treće točke putanje čime se dobiva poravnanje točaka u obliku glatkog pravca. Međutim, prethodno opisani proces raspoređivanja temeljen na translaciji točke na polovinu udaljenosti između prethodne i slijedne točke riješio je i problem naglih skretanja (prva pretpostavka fizičke izvedivosti putanje) ako se u procesu translatirane točke ažuriraju te se nove vrijednosti koriste za provjeru uvjeta ostalih točaka.



Slika 2.59. Grafički prikaz postupka razmještanja točaka

U konačnici, pomoću jednostavnog postupka translacije točaka temeljenog provjeri trigonometrijskih uvjeta dobiven je raspored točaka pomoću kojeg je moguće interpolirati glatku putanju postupcima prilagođavanja krivulje. Rezultati postupka prilagođavanja točaka su prikazani na slici 2.59. gdje su razmještene točke prikazane crvenim x znakovima.

Drugi korak interpolacije putanje je prilagođavanje krivulje (eng. curve fitting). Pri-



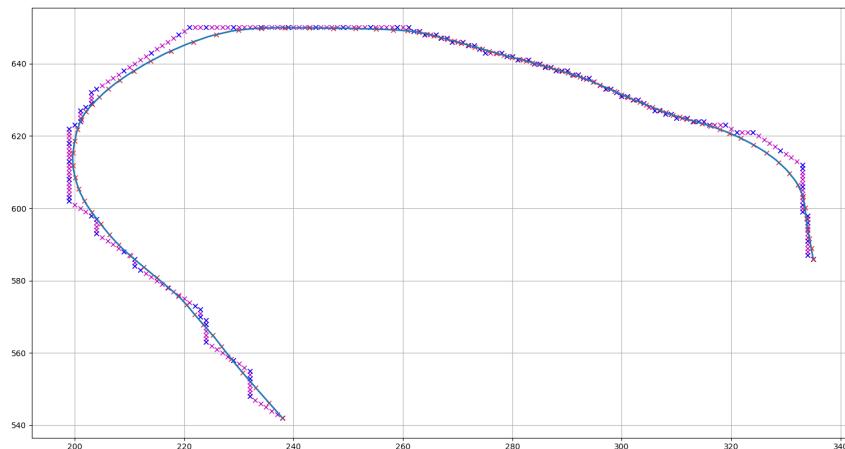
Slika 2.60. Rezultat razmještanja točaka

Iagođenje krivulje je matematička tehnika koja se koristi za pronalaženje krivulje koja najbolje odgovara skupu podataka. Cilj je konstruirati funkciju koja blisko prati zadane točke, često kako bi se omogućile predikcije ili interpolacija vrijednosti između tih točaka. Raspoređene točke putanje prosljeđuju se jednoj od funkcija prilagođavanja krivulje što definira parametar *optimization_method* iz launch datoteke. Metode prilagođavanja krivulje koriste se kako bi se kroz raspoređene točke interpolirana glatka izvediva putanja te nadomjestilo gubitak podataka o geografskim koordinatama zbog uzorkovanja koordinata mape prostora. Dostupne metode prilagođavanja krivulje preuzete su s Github repozitorija [4] te su ukratko opisane u nastavku.

Dubins krivulje predstavljaju najkraću putanju između dvije točke za vozilo koje se može kretati samo naprijed s konstantnim radijusom okretanja. Bezier krivulje su definirane kontrolnim točkama i koriste se za modeliranje glatkih i preciznih krivulja. Početna i završna točka definiraju krajeve krivulje, dok kontrolne točke utječu na njezin oblik. Polinomske krivulje koriste polinome za aproksimaciju ili interpolaciju podataka. One su korisne kada je potrebna glatka putanja koja prolazi kroz zadane točke ili kada je potrebno aproksimirati skup podataka. Reeds-Shepp krivulje su slične Dubins krivuljama, ali omogućuju kretanje vozila unatrag. Ovo ih čini pogodnima za planiranje putanja vozila koja se moraju kretati naprijed i unatrag kako bi dosegla cilj. Kubične krivulje koriste kubične polinome za modeliranje putanja. Ove krivulje omogućuju glatke i prirodne putanje. B-Spline krivulje su generalizacija Bezier krivulja koje koriste spline funkcije za modeliranje glatkih krivulja. Omogućuju lokalnu kontrolu nad oblikom krivulje, što znači da promjena jedne kontrolne točke utječe samo na dio krivulje. Rezultati

izgleda putanje različitih metoda prilagođavanja krivulje opisani su u poglavlju 2.6.5.

U trenutku kada jedna od metoda prilagođavanja krivulje izračuna konačno interpoliranu putanju, rezultati se prosljeđuju akcijskom serveru čime završava proces interpolacije. Primjer rezultata postupka interpolacije putanje prikazan je na slici 2.61. gdje su ružičastim x znakovima naznačene točke putanje D* lite algoritma, plavim x znakovima uzorkovane točke, a crvenim x znakovima razmještene točke putanje te plavom krivuljom prilagođena krivulja putanje dobivena pomoću Bezier krivulje.

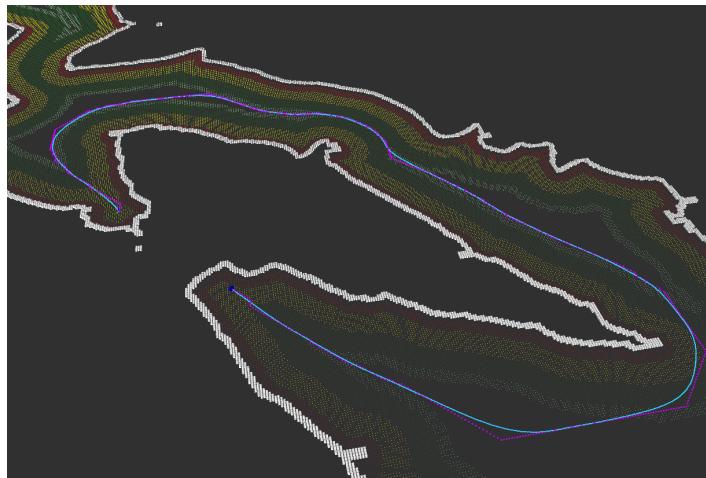


Slika 2.61. Rezultat interpolacije putanje

2.6.4. Objavljivanje rezultata putem action client i action server mehanizma komunikacije i prikaz rezultata u RViz2 vizualizacijskom alatu

Nakon završetka interpolacije putanje, u serverskom čvoru *path_planning server.py* potrebno je još izračunati duljinu putanje i procijenjeno vrijeme trajanja putanje. Duljina putanje računa se kao zbroj euklidskih udaljenosti između točaka putanje. Procijenjeno vrijeme trajanja putanje parametra *speed_limits* proslijedjenog iz launch datoteke. Naime, navedeni parametar sadržava informacije o ograničenju brzina u mjerenoj jedinici nautičkih milja za crvenu, žutu i zelenu zonu te željenu brzinu plovila u području bez ograničenja. Pod pretpostavkom da će plovilo voziti maksimalnim dopuštenim brzinama u zonama ograničenja izračunava se procijenjeno vrijeme trajanja putanje. Prvo se nautičke milje pretvaraju u metre po sekundi te se između svake dvije slijedne točke putanje izračunava procijenjeno vrijeme trajanja plovidbe kao euklidska udaljenost podijeljena sa maksimalnom dopuštenom brzinom u ovisnosti u kojoj se zoni točke nalaze. Ukoliko su točke u različitim zonama, računa se srednja vrijednost brzine ograničenja zona. Brzine u zonama putem parametra se također mogu i prilagoditi u ovisnosti o željenim postavkama plovidbe. Duljina putanje i procijenjeno trajanje plovidbe računaju se za putanje u interpoliranom i ne interpoliranom obliku.

Na kraju se koordinate interpolirane i neinterpolirane putanje pretvaraju u geografske koordinate globalnog koordinatnog sustava zajedno s informacijama o duljinama i procijenjenom vremenu trajanja putanje spremaju u polje rezultat akcije *StartGoalAction.action*. Navedena se akcija proslijedi akcijskom serveru čime završava proces planiranja putanje. Proces pretvorbe podataka u akcijskom klijentu u oblik pogodan za vizualizaciju putanje opisan je pri kraju poglavљa 2.6.3. Prikaz vizualizacije interpolirane i neinterpolirane putanje u Rviz2 vizualizacijskom alatu prikazan je na slici 2.62. dostupnih putem tema *path_pcd_points* i *raw_path_pcd_points*.



Slika 2.62. Vizualizacije interpolirane i ne interpolirane putanje u RViz2 alatu

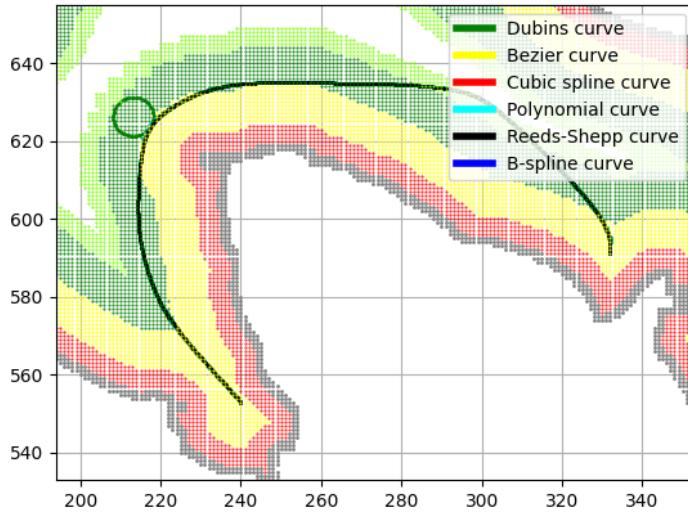
2.6.5. Rezultati i karakteristike konačne putanje u ovisnosti o parametrima procesa planiranja putanje

Ispitivanje algoritama prilagođavanja krivulje

Prije nego li se pristupi objašnjenju rezultata planiranja putanje plovila, potrebno je ispitati koji su od navedenih algoritama prilagođavanja krivulje optimalni za taj proces. Ispitani su algoritmi navedeni u poglavlju 2.61. što je prikazano na slici 2.63. Iz rezultata je vidljivo da su svi algoritmi osim Dubinsove krivulje pogodni za planiranje putanje plovila jer su točke interpolacije na većim udaljenostima te su postupkom razmještanja točaka izbjegnute nagle promijene smjera putanje. Kod optimizacije putanje korištena je Bezier krivulja, no s obzirom na to da su i ostale krivulje optimalne rezultati bi bili podjednaki.

Optimizacija putanje s obzirom na vrijednosti mape cijena

Hipoteza rada predstavljena u poglavlju 1.2. navodi da se opisane metode planiranja i interpolacije putanje plovila mogu primijeniti za planiranje putanje stvarnih plovila srednjih i manjih duljina ako je zadaća planiranja od polazišne do odredišne točke bez specifičnih namjena. Nadalje, osnovna je zadaća svakog algoritma planiranja putanje neovisno o modelu okruženja dovesti objekt za koji se putanja planira najkraćom izvedivom sigurnom rutom od polazišne do odredišne točke putanje. Prema navedenom, postavljeni kriteriji optimizacije putanje su sigurnost plovidbe, fizička izvedivost putanje te uniformnost kretanja u vidu brzina.



Slika 2.63. Rezultati procesa prilagođavanja krivulje

Kriterij fizičke izvedivost putanje podrazumijeva da je isplanirana putanja glatka i bez naglih promjena smjera, odnosno da su radijusi skretanja plovila relativno veliki te izvedivi velikoj većini plovila manjih i srednjih duljina. Kriterij uniformnosti kretanja postavlja uvjet na promjene brzine plovila koje mogu predstavljati sigurnosni i izvedbeni problem, stoga je zbog ovog kriterija uvedena sigurna zona. Utjecaj je sigurne zone na uniformnost kretanja putanje opisan u nastavku poglavlja. Kriteriji optimalnosti postavljeni su u okviru nekoliko pretpostavki. Prva pretpostavka je da se putanja planira od polazišne do odredišne točke sigurnom putanjom koja prolazi na većim udaljenostima od obale. Takvo što je moguće ako plovidba nije planirana za specifične namjene te ako kriteriji optimalnosti nisu duljina putanje i zahtjev za uštedom energenata. Nadalje, pretpostavka je da se plovilo može kretati većim brzinama od ograničenja brzina u crvenoj, žutoj i zelenoj zoni te da to i čini, ako propisi ne zahtijevaju drukčije, kako bi se minimiziralo vrijeme trajanja plovidbe. Pojmovi sigurne plovidbe i minimizacije trajanja često su posljedično povezani. Naime, plovilo se kroz sigurnu zonu i otvoreno more može kretati maksimalnom ili željenom brzinom plovidbe koje su redovito veće od maksimalnih brzina ograničenja zelene, žute i crvene zone. Područja bez ograničenja brzina su udaljena od obale minimalno 300 metara po zakonskim propisima te u njima ima manje potencijalnih sigurnosnih problema za plovidbu. Sigurnosni problemi su plitka područja te statične prepreke prirodnog ili ljudskog porijekla kojih je brojčano više u zonama bliže obali te nisu poznati algoritmu globalnog planiranja putanje. Također, na većim udaljenostima od obale vrijeme reakcije na nepredviđene čimbenike se povećava

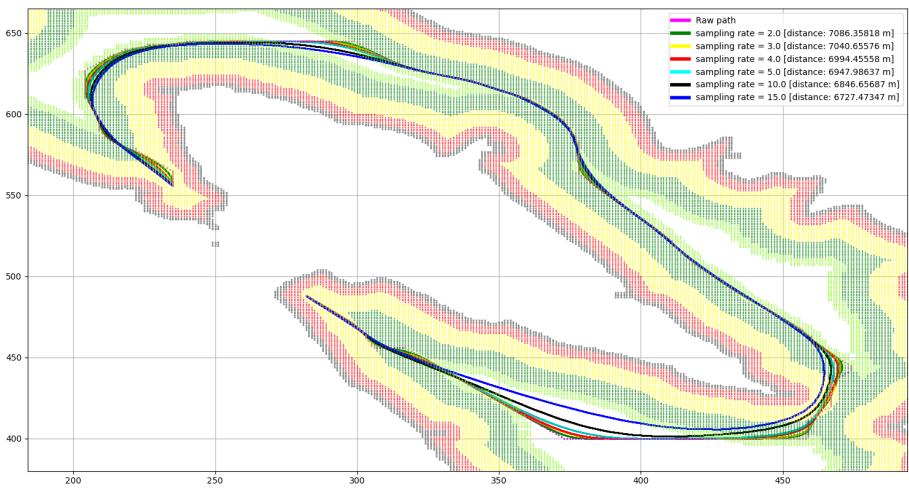
u odnosu na područja koja su bliže obali, što je detaljno objašnjeno prilikom postavljanja hipoteze rada. Zbog ranije navedenog, udaljavanjem putanje od obale smanjuje se vrijeme trajanja plovidbe te povećava vrijeme reakcije na nepredviđene čimbenike koji predstavljaju sigurnosne probleme što ispunjava navedene kriterije optimizacije. Kako bi planiranje putanje plovila ispunilo zahtjeve, putanja se modelira postavljanjem cijena zona udaljenosti od obale i prilagodbom parametara interpolacije putanje.

Prva odabrana lokacija za testiranje postavljenih zahtjeva je poluotok u blizini Krčkoga mosta sa specifičnim geografskim obilježjima u kojima je moguće prikazati utjecaj svih čimbenika procesa na izgled konačne putanje. Naime, polazišna točka putanje nalazi se u uvali Voz, a odredišna u uvali Peškera na suprotnim stranama poluotoka, iako ih dijeli samo nekoliko stotina metara kopna. Nadalje, od polazišne do odredišne točke putanja mora proći morskim kanalom s istaknutim rtovima Glavina i Voščica koje je potrebno zaobići te tjesnacem kod rta Bejavec nakon kojeg se prostire prostor šireg kanala i uvala Peškera.

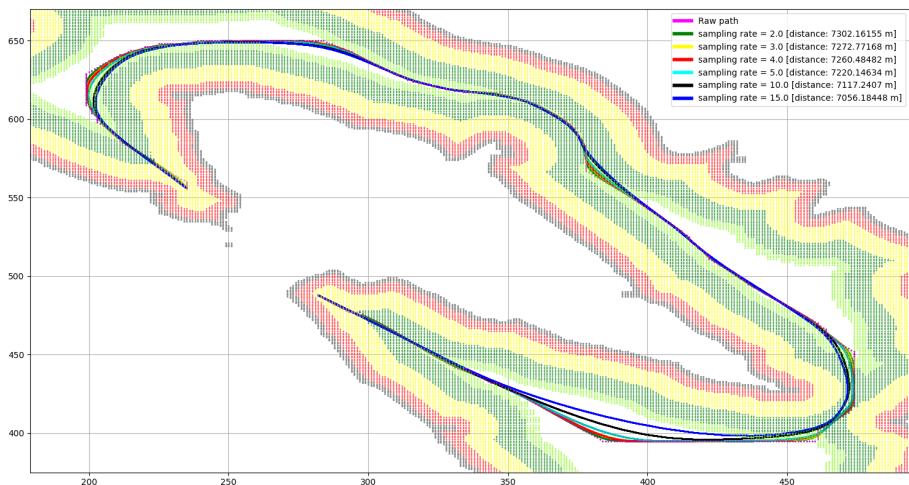
Pojam sigurna zona spomenut je više puta tijekom rada, no njegovo razumijevanje zahtjeva poznavanje koncepata interpolacije putanje. Naime, sigurna zona je zapravo uvedena zbog procesa interpolacije putanjom, tj. osiguravanja uniformnosti kretanja kroz zone. Proces interpolacije zaglađuje putanju čime je ispunjen kriterij optimizacije fizičke izvedivosti putanje, ali se gubi dio informacija o zonama (u procesu interpolacije se ne uzimaju u obzir cijene prostora). Zbog navedenog, ako putanja prolazi rubom zelenog ili žutog područja, kod promjene smjera putanje dolazi do neželjenih ishoda jer interpolirana putanja prolazi drugom zonom od njenog neinterpoliranog oblika. Takvo što zbog sigurnosti i udobnosti vožnje nije zadovoljavajuće jer plovilo skretanje započinje u zoni bez ograničenja brzine plovidbe te se potom kreće kroz zelenu zonu s ograničenjem brzine da bi skretanje završilo ponovno u zoni bez ograničenja. Takvo što nije niti praktično jer bi se prilikom skretanja moral smanjiti brzina plovidbe kako bi se poštivali zakonski propisi pa bi se na kraju skretanja brzina ponovno morala povećati. Drugačije rečeno, u takvima zonama uniformnost putanje u vidu promjene brzine nije zadovoljena. Nagle i kratkotrajne promjene brzine plovila također su štetne i po pogonski sustav i sustav prijenosnika plovila zato se takvo što nastoji izbjegći. Na slici 2.64. je pokazan ranije opisani problem, gdje putanje bez obzira na vrijednost parametra uzorkovanja

sampling_rate prolaze zelenom, odnosno žutom zonom rta Glavina i rta Bejavec, iako ostatak putanje koji prethodi i slijedi prolazi kroz druge zone, što zahtjeva prilagodbu brzine plovila. Razlog tome je što je sigurna zona postavljena na istu cijenu kao i zona bez ograničenja plovidbe te algoritam D* lite planira putanju u velikom dijelu prostora na rubu zelene zone zbog najmanjeg troška prolaska. Postupak razmještanja točaka navedene točke na rubu zelene zone razmješta u žutu zonu kod rta Glavina te točke putanje iz zelene zone u žutu zonu kod rta Berjavec što je preduvjet fizičke izvedivosti putanje. Dakle, postupak razmještanja točaka putanju čini glatkom i fizički izvedivom što je prvi postavljeni kriterij optimalnosti. Međutim, razmještanja točaka utječe na kriterij uniformnosti putanje u vidu brzina pa se može zaključiti da su navedeni kriteriji optimalnosti ovisni jedan o drugome. Sukladno navedenom, potrebno je obratiti pozornost prilikom procesa uzorkovanja putanje jer može posljedično utjecati na optimalnost konačne putanje. Naime, iako će kriterij fizičke izvedivosti putanje biti zadovoljen, on može dovesti do neispunjerenja kriterija uniformnosti putanje. Kako bi se navedeni problem uniformnosti izbjegao, uvodi se sigurna zona na udaljenosti od obale od 300 do 350 metara. Uvođenjem se sigurne zone postiže učinak pomicanja isplanirane putanje dalje od ruba zelene zone. Shodno navedenom iz slike 2.65. je vidljivo da je uvođenjem sigurne zone uspješno riješen problem prijelaza u žutu, odnosno zelenu zonu tijekom promjene smjera putanje plovila. Rezultati su zadovoljavajući i za velike vrijednosti parametra uzrokovavanja *sampling_rate* što je bitna činjenica jer sigurna zona omogućava postavljanje parametra uzrokovavanja i do većih vrijednosti prilikom ovakva postavke reljefa bez da parametar direktno utječe na ranije spomenute probleme prijelaska putanje u nepoželjna područja.

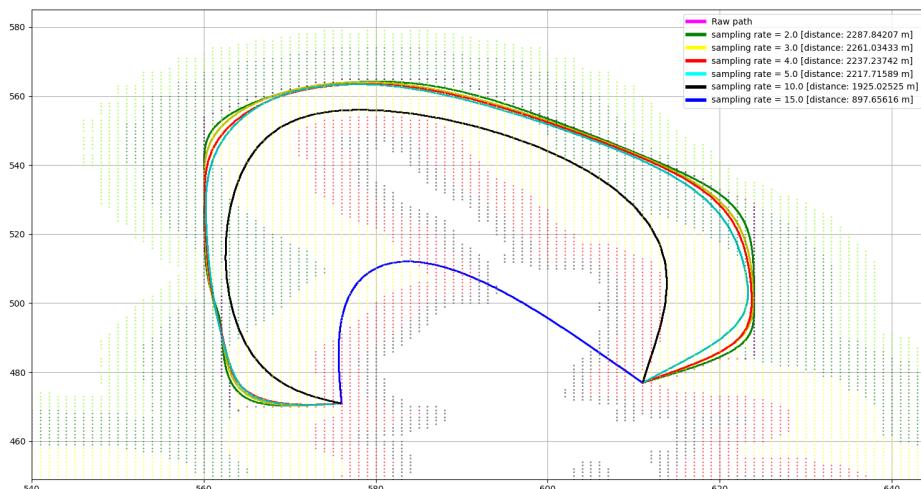
Kako bi se postiglo planiranje putanje koja većim dijelom prolazi kroz sigurnu zonu i zonu bez ograničenja, potrebno je postaviti podjednake cijene zelene i sigurne zone te višestruko veće cijene žute zone. Pravilo je da se vrijednost cijene crvene zone uvijek postavlja nekoliko puta veća od vrijednosti cijene žute ili zelene zone. Razlog tome je što crvena zona predstavlja područje opasnosti te bi putanja njome trebala prolaziti jedino prilikom uplovljavanja ili isplovljavanja pa se velikom vrijednosti cijene crvene zone upravo to i postiže. Postupkom pokušaja i pogrešaka dobiven je zaključak da točne vrijednosti cijena nisu značajne, već samo njihov omjer. Naime, ako se cijene zelene i žute zone postave na približno jednake vrijednosti koje su barem duplo veće od jedan (cijena prolaska zonom bez ograničenja) te se vrijednosti žute i zelene zone postave višestruko



Slika 2.64. Prikaz rezultata putanje uz različite vrijednosti parametra uzorkovanja i s cijenom sigurne zone 1



Slika 2.65. Prikaz rezultata putanje uz različite vrijednosti parametra uzorkovanja i s cijenom sigurne zone većom od 1



Slika 2.66. Utjecaj parametra uzorkovanja na izgled putanje

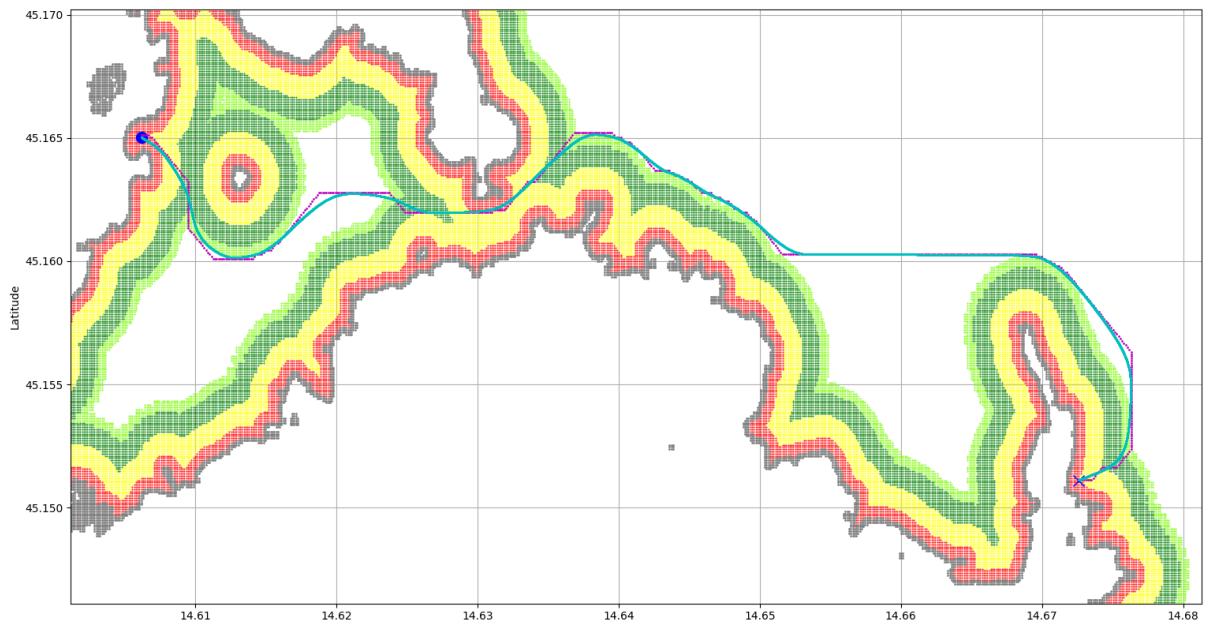
veće od toga, postiže se gotovo uvijek jednak rezultat. Takvo nešto je moguće zbog načina na koji računa D* lite algoritam, odnosno konačna putanja prolazi kroz područja s manjim cijenama. Nadalje, bitan razlog postavljanja cijena žute zone višestruko većih od cijena zelene zone je planiranje putanje kroz tjesnace. Naime, treba se osigurati da kroz tjesnac poput onog kod rta Bejavec u kojem ne postoje točke sigurne zone, putanja prolazi zelenom zonom koju preferira što se ovakvim omjerom cijena i postiže.

Zaključno, uvođenjem sigurne zone čija su cijene približno jednake onima u zelenoj zoni te postavljanjem cijena žute zone višestruko većih navedenih, moguće je isplanirati putanju koja zadovoljava sve kriterije optimalnosti.

Jedino što je preostalo ispitati je utjecaj parametra uzorkovanja na izgled putanje. Navedeni primjer nije pogodan za prikazivanje utjecaja parametra uzorkovanja jer sigurna zona osigurava da putanje uzorkovane velikim parametrima i dalje ispunjavaju uvjete. Kako bi se prikazali učinci parametra uzrokovana, odabранo je geografsko područje kod rta Šilo te su polazišna i odredišna točka putanje postavljene na suprotnim stranama rta. Dobiven je rezultat prikazan na slici 2.66. Iz slike 2.65. bi se dalo dobiti privid kako veći parametar uzorkovanja zapravo skraćuje konačno putanje uz ispunjenje svih kriterija optimalnosti, međutim primjer s navedene slike takvo što opovrgava. Putanje s vrijednošću parametra uzorkovanja 10 i 15 predstavljaju sigurnosni problem plovidbe jer prolazi blizu i kroz obalu. Iz slika 2.64. i 2.66. je vidljivo da parametar uzorkovanja utječe direktno na glatkoću putanje, odnosno veći parametar uzorkovanja osigurava veći radijus skretanja. Veći radijus skretanja posljedično znači širu prilagodbu različitim vrstama plovila. Takvo što je izuzetno korisno kako bi se planiranje putanje generaliziralo. S druge strane, manji parametar uzrokovana kao posljedicu ima manji radijus skretanja plovila te gubitak generalizacije. Uzveši u obzir sve prepostavke, odabran je parametar uzorkovanja 5 koji nije dovoljno velik kako bi utjecao na sigurnosne probleme poput parametara uzrokovana 10 i 15, a s druge strane osigurava veći radijus skretanja plovila što je temeljna prepostavka generalizacije algoritma.

Naposljetku, navedene zaključke o vrijednosti cijena i parametra uzorkovanja potrebno je ispitati na složenom primjeru kako bi se potvrdila ili opovrgnula njihova vjero-dostojnost. Za krajnji test ispitivanja uporabljivosti algoritma za globalno planiranje plovila odabранo je složeno geografsko područje oko rta Šilo povezano sa zaljevom Soline. Rt Šilo se pokazao kao zahtjevna geografska lokacija kod ispitivanja parametra uzorkovanja, a zaljev Soline karakterizira uzak pomorski tjesnac na ulazu kod rta Sulinj. Tjesnac u nazujoj točki ima razdaljinu manju od 300 metara, tj. sadrži samo točke žute zone što ga čini višestruko zahtjevnijim područjem za planiranje putanje od tjesnaca kod rta Bajec sa slike 2.65. Početna točka putanje postavljena je kod rta Šilo, a odredišna točka u zaljevu Soline u neposrednoj blizini otočića Školjić koji predstavlja dodatnu prepreku. Rezultati planiranja putanje u interpoliranom (svjetlo plava boja) i neinterpoliranom (ružičasta boja) obliku prikazani su na slici 2.67. Iz slike se uviđa razlog uvođenja interpolacije usporedbom izgleda putanja jer neinterpolirana putanja nije glatka pa posljedično i fizički izvediva. Nadalje, interpolirana putanja predstavlja zadovoljavajuće rezultate s obzirom na zahtjeve planiranja putanje i kriterije optimizacije, osim u području tjesnaca na izlazu iz zaljeva Soline što je očekivani rezultat. Putanja kroz navedeni tjesnac prolazi neposredno uz crvenu zonu te manjim dijelom i zalazi u nju. Ovaj problem nije moguće riješiti postavljanjem drugačijeg parametra uzrokovana niti drugačijih odnosa mape cijena jer su cijene zona pri trenutnim postavkama ujednačene i ne uzimaju u obzir potencijalni problem prolaska kroz tjesnac na manjim udaljenostima od obale. Pristup rješenju ukazanog problema bi bio uvođenje dodatnog parametra koji bi računao udaljenost od obale te s obzirom na parametar udaljenosti postavljao vrijednosti mape cijena. Navedenim pristupom, vrijednosti cijena zona ne bi bile ujednačene. Pri ovakvim postavkama mape cijena putanja bi prolazila sredinom uskih tjesnaca čime bi se riješio potencijalni sigurnosni problem prolaska putanje u blizini crvene zone.

Iako su dobiveni rezultati zadovoljavajući, primjer sa slike 2.67. pokazuje da i dalje postoje određeni nedostatci prilikom planiranja putanje plovila u područjima sa složenim reljefnim karakteristikama. Naime, potrebne su dodatne prilagodbe algoritma kako bi konačna putanja plovila kroz uske tjesnace i razvedena područja bila primjenjiva i za planiranje plovila u stvarnom okruženju. Međutim, rezultat sa slike 2.65. pokazuje da je algoritam ipak primjenjiv i u složenim okruženjima, ako ih ne karakteriziraju tjesnaci uži od 300 metara.



Slika 2.67. Završno testiranje planiranja putanje na području rta Šilo i zaljev Soline

3. Zaključak

Modelske informirano globalno planiranje putanja plovila (engl. Model-based path planning) se odnosi na metode planiranja putanja koje koriste model okruženja i dinamike plovila kako bi odredile optimalnu i efikasnu globalnu putanju plovila po različitim kriterijima. Matematički model okruženja uključuje geografsku mapu plovnog područja s istaknutim koordinatama obale koje naznačuju prepreku, koordinatama udaljenosti od obale, dubinama mora te preprekama na moru u što se ubrajaju manji otoci, hridi i signalizacijski objekti. Geografska mapa plovnog područja izrađena je obradom značajki nekoliko fotografija i HTML zapisa preuzetih s OpenStreetMap internetske stranice. Značajke obrađenih fotografija su raspoređene u ovisnosti o rubnih koordinatama te ih je shodno tome potrebno poravnati kako bi se stvorio ujednačen prikaz prostora u obliku kolaž mape geografskih koordinata. Metode obrade fotografije i poravnjanja geografskih koordinata temelje se na principu izrade lokalnog koordinatnog sustava. Lokalni koordinatni sustav proizlazi iz globalnog koordinatnog sustava i u osnovi se koristi za pretvorbu geografskih koordinata u metarsku skalu.

Kako bi se izrađena mapu geografskog prostora povezala s metodama postavljanja polazišne i odredišne točke te metodama planiranja putanje odabran je ROS2 okvir za izradu programske potpore. Također, usvojen je i princip crne kutije koji podrazumejava da različiti ROS2 paketi međusobno razmjenjuju informacije putem mehanizama samo sa sadržajem geografskih koordinata što posljedično zahtjeva pretvorbe koordinatnih sustava u svakoj logičkoj cjelini procesa. Koncept je usvojen zbog modularnosti koda, odnosno kako bi se svaki od navedenih paketa u budućim fazama projekta mogao zamjeniti s novijom naprednjom verzijom ili povezati s vanjskim sučeljima jer se informacije razmjenjuju u obliku geografskih koordinatama koje su standardne i prihvачene u velikom spektru programskih rješenja.

Postupak vizualizacije podataka temelji se također na pretvorbi geografskih koordinata u oblik pogodan za vizualizaciju u RViz2 alatu. Osim vizualizacije, RViz2 je i dijelom procesa postavljanja polazišne i odredišne točke putanje. Točke putanje se mogu postaviti na vizualiziranoj mapi geografskog prostora ili putem naredbe u Linux terminalu. Postavljanje polazišne i odredišne točke se vrši u svrhu globalnog planiranja putanje plovila kojim presjeda action client i action server ROS2 mehanizam komunikacije. Postavljene točke putanje klijentski čvor dostavlja serverskom čvoru koji računa putaju te klijentskom čvoru povratno dostavlja informacije o području pretrage algoritma te konačnu putanju u interpoliranom i neinterpoliranom obliku s pripadajućim informacijama o duljini putanje i predviđenom vremenu plovidbe. Konačne putanje se potom vizualiziraju u RViz2 alatu.

Proces planiranja putanje sastoji se od izračuna neinterpolirane putanje uz pomoć D* lite algoritma prilagođenog za rad na mapi cijena i procesa interpolacije putanje. Mapa cijena prostora predstavlja okolinu u obliku rešetke (eng. grid), gdje svaka komponenta obale ili zone udaljenosti od obale ima pridruženu cijenu koja označava trošak prolaska kroz tu komponentu. Međutim, mapa cijena ne posjeduje informacije o dinamičkim karakteristikama plovila te posljedično izgled putanje nije gladak, tj. fizički izvediv jer ima nagle promjene smjera, odnosno male radijuse skretanja. Kako bi se neinterpolirana putanja u konačnici učinila fizički izvedivom, koristi se proces interpolacije koji se sastoji od dva koraka. Prvi korak je raspoređivanja točaka pomoću detekcije naglih promjena smjera i uzrokovavanja krivulje određenim parametrom koji ima utjecaj na konačni izgled putanje. U osnovi procesa raspoređivanja točaka su jednostavne trigonometrijske funkcije za određivanje kutova između tri slijedne točke putanje pomoću kojih se postavljaju uvjeti pomicanja točke na novu poziciju po principima glatkoće putanje. Drugi korak je postupak prilagođavanja krivulje raspoređenih točaka kako bi se u konačnici generirala glatka, ujednačena i izvediva putanja plovila.

Hipoteza rada navodi da se opisane metode mogu primijeniti za planiranje putanje stvarnih plovila ako je zadaću planiranja plovila bez specifičnih namjena uz pravilno postavljene cijene prostora i parametara interpolacije. Navedena hipoteza je ispitana te je u rezultatima pokazano da je moguće isplanirati sigurnu izvedivu putanju od polazišne do odredišne točke putanje osim u karakterističnim slučajevima planiranja putanje kroz

tjesnace uže od 300 metara. Tvrđnja da je putanja izvediva za veliku većinu plovila manje i srednje duljine proizlazi iz činjenice da navedene putanje ne sadrže nagla skretanja, odnosno da su svi radijusi skretanja prihvatljivi i izvedivi za navedena plovila. S druge strane, pojam sigurnosti putanje u ovome trenutku ne može se nazvati tvrdnjom već pretpostavkom. Razlog tome je što izrađene mape ne sadržavaju informacije o dubinama mora i sigurnosnim prerekama pa se ne može sa sigurnošću utvrditi postoje li kakvi dodatni sigurnosti problemi koji kroz trenutnu mapu nisu zabilježeni. U konačnici, iz rezultata i opisa problema se može zaključiti kako trenutni algoritam planiranja putanje u ovoj fazi projekta nije primjenjiv za planiranje putanje plovila u stvarnome okruženju. Međutim, kroz pravilne prilagodbe mape cijena koje bi uključivale hidrografske informacije te parametar točne udaljenosti od obale pomoću kojeg bi se izbjegao sigurnosni problem planiranja putanje plovila kroz tjesnace uže od 300 metara, u budućim je fazama projekta moguće je izraditi funkcionalni algoritam za ispitivanje u stvarnome okruženju što i je krajnji cilj projekta modelski informiranog globalnog planiranja putanje plovila.

Literatura

- [1] Y. Wu, T. Wang, i S. Liu, “A review of path planning methods for marine autonomous surface vehicles”, *Journal of Marine Science and Engineering*, sv. 12, str. 833, 05 2024. <https://doi.org/10.3390/jmse12050833>
- [2] A. Sakai, D. Ingram, J. Dinius, K. Chawla, A. Raffin, i A. Paques, “Pythonrobotics: a python code collection of robotics algorithms”, 2018.
- [3] S. Koenig i M. Likhachev, “D* lite”, u *Proceedings of the National Conference on Artificial Intelligence*, 01 2002., str. 476–483.
- [4] ai winter, “python_motion_planning”, https://github.com/ai-winter/python_motion_planning, 2023.

Sažetak

Modelske informirano globalno planiranje putanje i upravljanje autonomnoga plovila

Enio Krizman

Predmet proučavanja ovoga je rada modelski informirano globalno planiranje putanja plovila, a odnose se na metode planiranja putanja koje koriste modele kako bi odredile globalnu putanju plovila. Programska potpora izrađena je u ROS2 okviru zbog zahtjeva modularnosti i komunikacije s vanjskim sučeljima. Mapa plovnog područja predstavlja matematički model okruženja i izrađena je obradom značajki nekoliko fotografija i njihovim poravnanjem u ujednačen prikaz geografskog prostora. Kako bi značajke obale i zona udaljenosti od obale mogле biti pretvorene u geografske koordinate, uveden je princip stvaranja lokalnog koordinatnog sustava u metarskoj skali. Navedeni princip se koristi i kod vizualizacije podataka u RViz2 alatu. Osim pri vizualizaciji podataka, RViz2 sudjeluje u procesu postavljanja polazišne i odredišne točke putanje. Postavljanje polazišne i odredišne točke putanje bitan je čimbenik proces planiranja putanje plovila pomoću klasičnog D* lite algoritma za računanje putanje na mapi cijena. Klasični D* algoritam bez modela dinamike plovila nije pogodan za planiranje bez prilagodbe jer putanja nije glatka, odnosno sadrži nagle promjene smjera. Shodno navedenom, izrađen je model interpolacije putanje postupkom raspoređivanja točaka i prilagođavanja krivulje. Hipoteza rada navodi da se opisane metode mogu primijeniti za planiranje putanje stvarnih plovila ako je zadaću planiranja putanje bez specifičnih namjena uz pravilno postavljene cijene prostora i parametara interpolacije. Navedena je hipoteza ispitana na mapama plovnog područja uz objašnjenje postupka postavljanja cijena prostora i uzorkovanja točaka putanje.

Ključne riječi: planiranje putanje, interpolacija putanje, izrada mape cijena, ROS2

Abstract

Model-informed global path-planning and autonomous vessel control

Enio Krizman

This paper addresses model-informed global path planning for vessels, specifically examining methods that use models to determine a vessel's optimal global path. The software support is developed within the ROS2 framework due to the requirements for modularity and communication with external interfaces. The navigational area map constitutes a mathematical model of the environment, generated by processing and integrating features from multiple images into a coherent representation of the geographical space. To convert the features of the shoreline and zones of distance from the shore into geographical coordinates, a principle of creating a local coordinate system on a metric scale was introduced. This principle is also used for data visualization in the RViz2 tool. In addition to data visualization, Rviz2 participates in the process of setting the starting and ending points of the path. Determining these points is a critical factor in the vessel path planning process using the classic D* Lite algorithm to calculate the path on a cost map. The classic D* algorithm, without the vessel dynamics model, is not suitable for planning without adjustments because the path is not smooth and contains sudden changes in direction. Consequently, a path interpolation model was created using a point distribution and curve fitting process. The hypothesis of the work posits that the described methods can be applied to the path planning of real vessels if the task is for general-purpose vessel planning with correctly set space costs and interpolation parameters. This hypothesis is tested on navigational area maps with an explanation of the process of setting space costs and path point sampling.

Keywords: path-planning, path interpolation, curve fitting, Cost map generation, ROS2

Privitak A: Programska potpora (the code)

Github repozitorij sa programskom potporom projekta izrade modelski informiranog globalnog planiranja putanje javno je dostupan na poveznici:

<https://github.com/kr1zzo/Model-Informed-Path-Planning>

A GitHub repository with software support for the project on model-informed global path planning is publicly available at the link:

<https://github.com/kr1zzo/Model-Informed-Path-Planning>