Complete coverage of large seabed areas using sonar data-based path planning for an autonomous marine vehicle

Kapetanović, Nadir

Doctoral thesis / Disertacija

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:168:941026

Rights / Prava: In copyright/Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-11-28**

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Nadir Kapetanović

COMPLETE COVERAGE OF LARGE SEABED AREAS USING SONAR DATA-BASED PATH PLANNING FOR AN AUTONOMOUS MARINE VEHICLE

DOCTORAL THESIS

Zagreb, 2023

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Nadir Kapetanović

COMPLETE COVERAGE OF LARGE SEABED AREAS USING SONAR DATA-BASED PATH PLANNING FOR AN AUTONOMOUS MARINE VEHICLE

DOCTORAL THESIS

Supervisors: Professor Nikola Mišković, PhD
Associate professor Adnan Tahirović, PhD

Zagreb, 2023

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Nadir Kapetanović

PLANIRANJE PUTANJE AUTONOMNOGA PLOVILA ZASNIVANO NA SONARSKIM PODACIMA U SVRHU POTPUNOGA PREKRIVANJA VELIKIH POVRŠINA MORSKOGA DNA

DOKTORSKI RAD

Mentori: prof. dr. sc. Nikola Mišković
izv. prof. dr. sc. Adnan Tahirović

Zagreb, 2023.

DOCTORAL THESIS is written at the University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Control and Computer Engineering.

Supervisors: Professor Nikola Mišković, PhD
Associate professor Adnan Tahirović, PhD

DOCTORAL THESIS has:196pages

Dissertation No.: _____

About the Supervisors

Nikola Mišković is a Full Professor at University of Zagreb, Faculty of Electrical Engineering and Computing where he teaches courses related to control theory and marine robotics. He received his diploma and Ph.D. degrees in electrical engineering from the University of Zagreb, Faculty of Electrical Engineering and Computing (FER), Zagreb, Croatia, in 2005 and 2010, respectively.

From July 2005 he is working at the Department of control and computer engineering at FER. He was a visiting researcher at the Consiglio Nazionale delle Ricerche in Genoa, Italy (in 2008). In 2016 he was promoted to Associate Professor, and became Full Professor in 2019. He is the Head of Laboratory for Underwater Systems and Technologies (LABUST). He participated in 15 European projects (Horizon Europe, H2020, FP7, DG-ECHO, INTERREG) out of which he coordinated FP7 CADDY, focussing on the development of the first underwater robot for interaction with divers; H2020 aPad, devoted to commercialization of an autonomous surface vehicles developed in LABUST, and H2020 EXCELLABUST devoted to increasing LABUST excellence in marine robotics. He also participated in 4 Office of Naval Research Global (ONR-G) projects (coordinated 3), 2 NATO projects, and 7 national projects (coordinated 3). He published more than 70 papers in journals and conference proceedings in the area of navigation, guidance and control, as well as cooperative control in marine robotics.

Prof. Mišković is a member of IEEE (president of Chapter for Robotics and Automation of the Croatian Section from 2016 to 2017), IFAC (member of the Technical Committee on Marine Systems) and Centre for Underwater Systems and Technologies (vice-president since 2010).

In 2020 Prof Nikola Mišković was awarded the IEEE Croatia Section Award for Outstanding Engineering Contribution for exceptional engineering contribution in the field of marine robotics, particularly innovative underwater robotic systems and autonomous surface vehicles. He received the annual State science award for 2015, awarded by the Croatian Parliament and in 2013 he received the young scientist award "Vera Johanides" of the Croatian Academy of Engineering (HATZ) for scientific achievements.

Adnan Tahirović is an Associate Professor at the Department of Automatic Control and Electronics of the Faculty of Electrical Engineering, University of Sarajevo, Bosnia and Herzegovina. He graduated and obtained his M.Sc. degree from the Faculty of Electrical Engineering, University of Sarajevo, in 2006, from which he received the Golden Plaque the best student of generation award.

In 2011 he was awarded a Ph.D. degree in Information Technology by the Department of Electronics and Information, Politecnico di Milano, Italy. He was a researcher at NASA JPL, Pasadena, USA, in 2010, and Imperial College London, in 2011. Since 2018 he also holds a visiting research position at Imperial College London.

Prof. Tahirovic is a Head of Laboratory for Collaborative Artificial Intelligence and Control Systems at the Faculty of Electrical Engineering, University of Sarajevo, which covers fundamental research of optimal control theory, collaborative artificial intelligence and multiagent systems. He was also a co-director of the MORUS project (NATO SPS), which was conducted in collaboration with the University of Zagreb, Faculty of Electrical Engineering and Computing. He currently participates in two EU projects. The goal of the first project (AeroSTREAM) is to establish long term cooperation between HEIs from widening countries (Croatia and Bosnia and Herzegovina) and leading institutions from Member states (Spain, Czech Republic, and the Netherlands), with special attention to stimulated scientific excellence and innovation capacity. The second project (MARBLE) aims to prepare a joint master dedicated to maritime robotics in blue economy, and it includes partner universities and stakeholders from Italy, Greek, Montenegro, Croatia and Bosnia and Herzegovina.

His research interests include nonlinear control theory, motion planning in mobile robotics, autonomous and multiagent systems, as well as control theory applications in the domain of artificial intelligence and computational neuroscience.

O mentorima

Nikola Mišković redoviti je profesor na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu gdje predaje kolegije vezane uz teoriju automatskog upravljanja i pomorsku robotiku. Diplomirao je i doktorirao u polju elektrotehnike na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva (FER) 2005. odnosno 2010. godine.

Od srpnja 2005. godine radi na Zavodu za automatiku i računalno inženjerstvo FER-a. Bio je gostujući istraživač na Consiglio Nazionale delle Ricerche u Genovi, Italija (u 2008. godini). 2016. godine izabran je u zvanje izvanrednog profesora, a 2019. u zvanje redovitog profesora. Voditelj je Laboratorija za podvodne sustave i tehnologije (LAPOST). Sudjelovao je u 15 europskih projekata (Horizon Europe, H2020, FP7, DG-ECHO, INTERREG) od kojih je koordinirao FP7 CADDY, projekt fokusiran na razvoj prvog podvodnog robota za interakciju s roniocima; H2020 aPad, posvećen komercijalizaciji autonomnih površinskih vozila razvijenih u LABUST-u, i H2020 EXCELLABUST posvećen povećanju LABUST-ove izvrsnosti u pomorskoj robotici. Također je sudjelovao u 4 projekta Office of Naval Research Global (ONR-G) (od kojih 3 kao koordinator), 2 NATO projekta i 7 nacionalnih projekata (koordinator 3). Objavio je više od 70 radova u časopisima i zbornicima konferencija u području navigacije, vođenja i upravljanja, te kooperativnog upravljanja u pomorskoj robotici.

Prof. Mišković član je stručnih udruga IEEE (predsjednik Odjela za robotiku i automatizaciju Hrvatske sekcije od 2016. do 2017.), IFAC (član Technical Committee on Marine Systems) i Centra za podvodne sustave i tehnologije (dopredsjednik društva od 2010.).

Godine 2020. dobitnik je Nagrade Hrvatske sekcije IEEE za izniman inženjerski doprinos (IEEE Croatia Section Outstanding Engineer Award) za izniman inženjerski doprinos u području pomorske robotike, posebice inovativnih podvodnih robotskih sustava i autonomnih površinskih vozila. Godine 2013. primio je nagradu Hrvatske akademije tehničkih znanosti „Vera Johanides" mladom znanstveniku za uspjehe u području istraživanja, te je nagrađen godišnjom Državnom nagradom za znanost za 2015. godinu.

Adnan Tahirović izvanredni je profesor na Zavodu za automatiku i elektroniku Elektrotehničkog fakulteta Sveučilišta u Sarajevu, Bosna i Hercegovina. Diplomirao je i stekao zvanje magistra znanosti na Elektrotehničkom fakultetu Sveučilišta u Sarajevu 2006. godine, s kojeg je dobio nagradu Zlatna plaketa za najboljeg studenta generacije.

Godine 2011. doktorirao je u oblasti informatičke tehnologije na Zavodu za elektroniku i informatiku na Politecnico di Milano, Italija. Bio je istraživač na NASA JPL, Pasadena, SAD, 2010. i Imperial College London, 2011. Od 2018. također je gostujući istraživač na Imperial College London.

Prof. Tahirović je voditelj Laboratorija za kolaborativnu umjetnu inteligenciju i sustave upravljanja na Elektrotehničkom fakultetu Sveučilišta u Sarajevu, koji pokriva temeljna istraživanja teorije optimalnog upravljanja, kolaborativne umjetne inteligencije i multiagentnih sustava. Bio je i suvoditelj projekta MORUS (NATO SPS) koji se provodio u suradnji sa Sveučilištem u Zagrebu, Fakultetom elektrotehnike i računarstva. Trenutno sudjeluje u dva EU projekta. Cilj prvog projekta (AeroSTREAM) je uspostaviti dugoročnu suradnju između visokih učilišta iz zemalja širenja (Hrvatske i Bosne i Hercegovine) i vodećih institucija iz država članica (Španjolska, Češka i Nizozemska), s posebnim osvrtom na poticanje znanstvenih sposobnost izvrsnosti i inovacija. Drugi projekt (MARBLE) ima za cilj pripremiti zajednički master posvećen pomorskoj robotici u plavoj ekonomiji, a uključuje partnerska sveučilišta i dionike iz Italije, Grčke, Crne Gore, Hrvatske i Bosne i Hercegovine.

Njegovi znanstveni interesi uključuju teoriju nelinearnog upravljanja, planiranje kretanja u mobilnoj robotici, autonomne i multiagentne sustave, kao i primjene teorije upravljanja u domeni umjetne inteligencije i računalne neuroznanosti.

Acknowledgements

I would like to express my deepest gratitude to my mentors Professor Nikola Mišković and Associate Professor Adnan Tahirović for their unwavering support and optimism during my PhD studies. In particular, I'd like to thank Adnan for inviting me to work on MORUS project, thus creating an opportunity for me to enroll in doctoral studies at FER. Furthermore, I'd like to thank Nikola for inviting me to join his team in Zagreb and immensely improve my technical, research, but also project management skills. Whenever I met an obstacle, they helped me see the bigger picture, see the problem from another perspective, and finally solve it.

I am also thankful to all current and former members of LABUST for all their support and valuable discussions. I would like to especially thank Đula Nađ, Milan Marković and Antonio Vasilijević for making so many strenuous sea trials easier with their calm attitude and humor. Furthermore, I am grateful to Ivana who showed me the ropes of the management side of research projects and became a dear friend in the process. Many thanks to Anja Babić, Filip Mandić and Ivan Lončar for all cross-project collaborations as well as philosophical and existential discussions, most notably on our way to, during, and on our way back from sea trials. I would like to extend my sincere thanks to Professor Zdenko Kovačić, Nikica Kokir, Martin Oreč, Kristijan Krčmar, Matej Fabijanić, Allan Badian for close collaboration on HEKTOR project for the past three years.

I'm grateful beyond words to my mom, dad and sister for their unconditional love and encouragement in all my life endeavors. Without them, all this would not have been possible.

Many thanks to my friends Naida, Darko and Eldar, whom I think of as my second source, for always being there for me, helping me grow personally and navigating sometimes tumultuous waters of life.

Special thanks to my partner Roland for sticking with me through thick and thin of my professional and personal life, and his support and constructive perspective on everything I do.

Finally, I would like to thank all people who have helped me in any way during this process, whom I might have forgotten to mention.

Abstract

Efficient mapping of an unknown large-scale marine area using a side-scan sonar on-board an autonomous marine vehicle is often of great importance. It might also be important to scan parts of an area in more detail and from more than one side. In contrast to the standard offline static coverage problem solution for side-scan sonar missions based on the overlapping all across-track sonar swaths in a lawnmower pattern, several online sonar-data driven coverage path planning algorithms are proposed in this thesis. The proposed algorithms provide a coverage solution based on local information gain computed from the side-scan sonar data during the mission execution, which is then exploited for replanning. In addition, the proposed solution takes into account the coverage path length/time to ensure equal information content about arbitrarily defined interesting objects as in case of the standard lawnmower with much less resources needed to complete the same mission.

One dynamical programming-based and three decision making-based coverage path (re)planning algorithms are proposed herein. Their performance was thoroughly examined in a grid-like coverage area cost map simulation over a whole range of coverage mission parameters and randomly generated seafloor configurations to gain statistical metrics of their behavior. Also, their upper and lower performance bounds are mathematically modelled w.r.t. some mission parameters, and were validated by the thousands of simulation runs. Moreover, an existing realistic 3D simulation environment was extended by the above-mentioned coverage path planners, mission controller, and sonar data processing modules to gain further insights into the interplay of coverage path planning, feasible path interpolation, vehicle dynamics and control.

Keywords: autonomous underwater vehicles, coverage path planning, sensor data-driven coverage, side-scan sonar, image processing, anomaly detection, target detection, saliency

Planiranje putanje autonomnoga plovila zasnivano na sonarskim podacima u svrhu potpunoga prekrivanja velikih površina morskoga dna

Učinkovito mapiranje velikih nepoznatih područja morskoga dna korištenjem bočno skenirajućeg sonara (engl. *side-scan sonar*) i autonomnoga pomorskog vozila je često od velikog značaja. Također, često je bitno detaljnije snimiti zanimljive dijelove tog područja i to s više od jedne strane. Predloženo je nekoliko *online* metoda prekrivanja područja zasnovanih na sonarskim podacima. Razvijene metode su poređene sa standardnim *offline* rješenjem problema prekrivanja sonarom, tzv. uzorkom kosilice koji sve objekte neadaptivno snima sa obje strane bez obzira da li uopće postoje ikakvi objekti na morskome dnu, i ako postoje da li su zanimljivi za trenutnu misiju. Za razliku od toga, predloženi algoritmi nude rješenje problema (re)planiranja prekrivanja područja zasnovano na sonarskim podacima u tijeku misije. Nadalje, oni u dosta velikom području vrijednosti parametara misije nude jednako informativnu snimku interesantnih objekata kao i neadaptivni pristup uz znatno skraćenje trajanja misije/dužine putanje prekrivanja.

Jedan algoritam zasnovan na dinamičkom programiranju i tri algoritma zasnovana na heurističkom donošenju odluka su predloženi u ovom radu. Njihove performanse su detaljno ispitane na matričnoj mapi cijena prekrivanja i to za širok opseg vrijednosti parametara misije i slučajno generarirane konfiguracije morskoga dna kako bi se dobio uvid u statistiku ponašanja navedenih algoritama. Također, njihove gornje i donje granice performansi su matematički opisane u ovisnosti od parametara misije, te su validirane tisućama provedenih simulacija. Štoviše, postojeće realistično 3D simulacijsko okruženje je prošireno sa gore navedenim planerima prekrivanja, kontrolerom misije, i modulom obrade sonarske slike kako bi se stekao još i bolji uvid u međusobne utjecaje modula planiranja putanje prekrivanja, interpolacije izvedivih putanja, te dinamike i upravljanja vozilom.

Glavni cilj istraživanja ovog rada je bio razvoj vremenski učinkovitog algoritma planiranja prekrivanja za autonomna podvodna vozila korištena za misije mapiranja bočno skenirajućim sonarom sa posebnim naglaskom na dugotrajni rad, tj. velika morska područja površine preko $1km^2$.

Doktorski rad podijeljen je na osam poglavlja. Prvo poglavlje („1. Introduction") daje uvod u motivaciju istraživanja, definiciju problema koji se treba riješiti, te pregled doprinosa teze i kratak opis sadržaja poglavlja.

Uvod u teoriju podsustava navigacije, vođenja i upravljanja kod autonomnih pod-

vodnih vozila uz poglavit osvrt na modelski prediktivno upravljanje dani su u drugom poglavlju („2. Navigation, Guidance and Control of Marine Vehicles"). NGC sustav je odgovoran za usmjeravanje pokretačkih sila i stabilizaciju vozila duž željene putanje. Sustav NGC mora definirati putanju u stvarnom vremenu kako bi dosegao zadani cilj i usmjerio vozilo duž željene putanje. Sustav za navođenje generira putanju za postizanje cilja i željene naredbe za upravljanje vozilom kako bi se ostvarila putanja u stvarnom vremenu. Sustav upravljanja vozilom prima naredbe za upravljanje od sustava za navođenje i upravlja vozilom da slijedi željeni položaj bez obzira na sve smetnje. Navigacija, navođenje i kontrola autonomnih plovila izazovan je zadatak, uglavnom zbog sprege između stupnjeva slobode, hidrodinamičkih učinaka i vrlo nepredvidivih utjecaja na okoliš. Trošak rada na moru i postavljanja za različite tipove autonomnih pomorskih vozila je visok, kao i rizik od gubitka vozila tijekom rada. Kao posljedica toga, potrebni su dobri matematički modeli za ispravno testiranje i simulaciju NGC sustava prije. Modeliranje senzora i vanjskih poremećaja mora ispravno približiti izazovnu i stohastičku prirodu podvodnog okruženja.

Pokazalo se da modelsko prediktivno upravljanje (MPC) ima dobre performanse kada se koristi za planiranje kretanja podaktuiranog autonomnog pomorskog vozila. Njegova glavna prednost kao metode planiranja gibanja je da generira kinematički (ili čak dinamički, ovisno o modelu koji se koristi) izvedive putanje, za koje optimizira upravljački signal(e) na takav način da dana ograničenja stanja sustava i upravljanja su upoznati. Ova metoda uzima u obzir model, troškovnu funkciju i ograničenja tijekom procesa optimizacije upravljačkog signala, za razliku od drugih metoda kao što su PID i LCF kontrola, koje se ne bave eksplicitno ograničenjima, te ih zadovoljavaju eventualnim zasićenjem generiranog upravljačkog signala. Ovo *a-posteriori* zasićenje može dovesti do značajnog pogoršanja performansi sustava, pa čak i nestabilnosti sustava u nekim slučajevima.

Modularni MPC okvir za opću upotrebu razvijen je korištenjem softvera otvorenog koda ACADO alata i integriran je s ROS-om. Ovisno o problemu upravljanja, model sustava, ograničenja, funkcije cijene i parametri algoritma mogu se jednostavno mijenjati. Implementacija MPC okvira u stvarnom vremenu postignuta je za problem slijeđenja linije koji je obrađen u ovom poglavlju. Rezultati simulacije i eksperimenta pokazuju dobre performanse MPC regulatora u usporedbi s PID regulatorom. Glavna prednost MPC nad PID regulacijom u ovoj perspektivi je njegova mogućnost da se koristi kao opći okvir za optimizaciju regulacije.

Treće poglavlje ("3. Sonar Data-based Coverage Path Planning Algorithms") donosi opis gore spomenutih algoritama prekrivanja baziranih na sonarskim podacima. Njihove performanse su poređene sa klasičnim uzorkom kosilice (LM) po pitanju trajanja

misije/ukupne dužine putanje prekrivanja, ukupnog vremena (re)planiranja, te statis-
tičkim pokazateljima ponašanja algoritma za dane parametre misije. Misije praćenja
i istraživanja mora često uključuju kartiranje morskog dna temeljeno na sonaru. To
uključuje istraživanje biosfere, istraživanje podvodnih arheoloških nalazišta, pomorsku
sigurnost i mnoge druge primjene. Trenutačno se misije sonara za bočno skeniranje
izvode pomoću tzv. *towfisha*, korištenjem daljinski upravljanih ronilica (ROV), ili au-
tonomnih podvodnih vozila (AUV). Pokretanje privezane teglice s broda zahtijeva an-
gažiranje broda i njegove posade: operatera towfisha, operatera bočnog skeniranja i
naravno stručnjaka u znanstvenom području za koje se provodi istraživanje. Stoga,
bilo bi mnogo praktičnije i jeftinije postaviti AUV i pustiti ga da samostalno skenira
određeno područje. Trebalo bi prikupiti više informacija o dijelovima područja
pokrivenosti koje smatra zanimljivim za trenutnu istraživačku misiju i podatke općeg
istraživanja niže rezolucije o drugim dijelovima područja.

Sljedeći korak prema gore navedenom cilju istraživanja bio je dizajn algoritma za
rješavanje problema online 2D pokrivenosti. Ideja je započeti misiju bez preklapanja
sonarnih opsega, tj. s LM trakama dvostruko širim od dometa sonara. Tijekom misije,
u slučaju da algoritam detektira nešto zanimljivo u traci LM kojom se trenutno prolazi,
algoritam za planiranje staze pokrivanja ponovno planira ostatak misije kako bi sonifi-
cirao te zanimljive objekte i sa suprotne strane, prikupljajući tako više informacija o
morfologiji od zanimljivih objekata.

Glavni doprinosi predloženih algoritama su: (1) CPP algoritmi zasnovani na di-
namičkom programiranju i heuristici zasnovani na online bočnom skeniranju sonara
za nepoznate terene koji vjerojatno sadrže relativno malo zanimljivih dijelova. Algo-
ritam pokriva područje u najboljem slučaju s dvostruko kraćim putem pokrivenosti
od klasičnog manevra pokrivenosti LM pokrivenosti statičkim statičkim preklapanjem
svih opsega sonara (odavde skraćeno kao metoda CL-CPP), (2) analitičke gornje i donje
granice performansi gore spomenutih CPP algoritama, potvrđenih opsežnim simulaci-
jama varijacija parametara misije, (3) statistička analiza performansi predloženih CPP
algoritama u odnosu neprilagodljivi LM manevar pokrivenosti sa statičkim prekla-
panjem svih opsega sonara (CL-CPP metoda), i (4) metodologija za približan odabir
parametara misije istraživanja na temelju topologije područja pokrivenosti i očekivane
vjerojatnosti nailaska na zanimljive objekata u datom području. Ovo može biti od ve-
like koristi inženjerima sustava koji provode istraživačke misije s pomorskim vozilom.

Četvrto poglavlje ("4. Object detection in side-scan sonar imagery") daje pregled
metoda detekcije anomalija, tj. generički definiranih zanimljivih objekata na morskome
dnu koji se po bilo čemu razlikuju od okoline koja je većinom samo ravno pješčano dno.

S porastom sonarskih sustava visoke rezolucije dolazi i potreba za pohranjivanjem

i/ili obradom ogromnih količina podataka. Slanje tih podataka u pokretu na neku centraliziranu zemaljsku stanicu na obradu općenito nije nepremostiv problem za zemaljska i zračna vozila budući da je WiFi veza moguća. U podvodnom okruženju, međutim, komunikacija je uvelike ograničena činjenicom da su elektromagnetski valovi jako prigušeni čak i na malim udaljenostima, dok u isto vrijeme akustične komunikacije imaju tako nisku propusnost da je slanje ogromnih količina sonarskih podataka u letu još uvijek nemoguće. Zbog toga se svi podaci sonara pohranjuju na ugrađeni tvrdi ili solid state disk kada se koriste AUV-ovi, što omogućuje samo offline obradu sonarskih slika, bilo od strane ljudskog operatera ili nekog algoritma za obradu slike. U oba slučaja podaci se ne mogu online analizirati tijekom misije, tako da ljudski operater ili algoritam (ponovnog) planiranja misije ne mogu detaljnije pregledati neke zanimljive objekte čim se pojave u podacima sonara.

Uzimajući u obzir pingove bočnog skeniranja naslagane zajedno (u takozvanom "vodopadnom prikazu") kao sliku u sivim tonovima, u ovom su poglavlju implementirane i testirane različite metode otkrivanja slike anomalija/istaknutosti. Ove metode, osim detekcije anomalija u podacima sonara bočnog skeniranja i vidljivosti temeljene na grafikonu (GBVS), gotovo su se isključivo koristile za prirodne slike, metoda istaknutosti temeljena na kontrastu, Itti-Kochova istaknutost i Simpsal istaknutost kao pojednostavljenje Itti-Koch metode. Gore spomenute metode su prvo uspoređivane u Matlabu na skupu od 1500 skupa podataka simuliranih slika sonara bočnog skeniranja koji sadrže jedan i više objekata različitih veličina i na različitim položajima. Metoda otkrivanja anomalija prethodno primijenjena na sonarne slike imala je najbolju preciznost i performanse prisjećanja na skupu podataka simuliranih sonarskih slika. Te su performanse zatim potvrđene na stvarnom skupu podataka od 500 slika sonara bočnog skeniranja, a i performanse preciznosti prisjećanja su bile zadovoljavajuće.

Peto poglavlje ("5. Simulation environment for coverage planning algorithms validation") opisuje razvijene module integrirane u postojeći realistični 3D simulator podvodnih autonomnih vozila kako bi se mogli validirati simulacijski rezultati zasnovani na matričnim mapama cijena prekrivanja, te analizirati međusobni utjecaji modula za planiranje prekrivanja, generiranje izvedivih trajektorija, upravljanje i obradu sonarskih podataka.

Algoritmi temeljeni na sonarnim podacima, naime CL-CPP, BA-CPP i OPTA-CPP, spomenuti u poglavlju 3, implementirani su u okviru ROS simulacijskog okruženja zajedno s algoritmima za detekciju anomalija sonara bočnog skeniranja spomenutim u poglavlju 4. Ovo poglavlje opisuje integraciju ovih modula s UUV simulatorom otvorenog koda koji je korišten kao osnova za validacijsko okruženje simulacije. Opisuje se UUV simulator, implementacija gore spomenutih CPP algoritama kao i modul kon-

trolera misije pokrivenosti. Dati su primjeri misija pokrivenosti korištenjem CL-CPP i OPTA-CPP algoritama zajedno s usporedbom performansi.

Autonomna vozila koja su korištena za eksperimentalne validacije nekih od gore navedenih podsustava opisana su u šestom poglavlju ("6. Systems"). Autonomna pomorska vozila igraju važnu ulogu u današnjim misijama kartiranja i istraživanja morskog dna i ovaj će se trend samo povećati u nadolazećim godinama. Ovo poglavlje donosi tehničke opise različitih vozila koja se koriste u studijama slučaja iz stvarnog svijeta predstavljenim u posljednjem poglavlju ove teze. Ova vozila su: LAUV Lupis koje je naš istraživački laboratorij nabavio od tvrtke OceanScan; ASV PlaDyBath, potpuno razvijen od strane LABUST-a u okviru projekta BLUEMED; ASV Korkyra, razvijen u okviru projekta HEKTOR, kao značajno veća verzija ASV PlaDyBath; i hibridni ROV/AUV Europe, koji su razvili naši kolege iz CNR-a, Genova.

Sedmo poglavlje ("7. Case studies") donosi pet eksperimentalnih studija vezanih za upravljanje pomorskim vozilima u cilju praćenja linijske putanje, lokalizaciju, snimanje područja nepravilnog oblika, snimanje podvodnih arheoloških nalazišta s više vrsta vozila i integracijaraznorodnih podataka, te validacija performansi algoritama obrade sonarskih slika sa hardverom u petlji.

U ovom su poglavlju predstavljene eksperimentalne studije slučaja različitih podsustava, tj. kontrolera za modelski prediktivno praćenje linije, modula za podvodnu lokalizaciju temeljenog na proširenom Kalmanovom filtru, i modula za obradu sonarnih slika bočnog skeniranja sa hardverom u petlji. Također su opisani rezultati opsežnih ispitivanja s autonomnim pomorskim vozilima koja se koriste u limnološkim i podvodnim arheološkim primjenama. Iskustvo autonomnog operatera stečeno tijekom ovih ispitivanja imalo je značajan utjecaj na dizajn predloženih algoritama za planiranje staze pokrivenosti, uključujući mnoga ograničenja upotrebe sonara za bočno skeniranje kao senzora za mapiranje. Također je rezultiralo značajnim skupom podataka - slikama sonara bočnog skeniranja koji se kasnije koristio za provjeru performansi algoritama za obradu sonarske slike.

Temeljem upravljačkih algoritama i metodologija za validaciju algoritama razvijenih unutar doktorata izdvojena su tri znanstvena doprinosa:

1. Algoritam planiranja prekrivanja velikih površina morskoga dna korištenjem autonomnoga plovila zasnovan na sonarskim podacima.

2. Simulacijsko okruženje za testiranje i ocjenjivanje učinkovitosti algoritma planiranja prekrivanja zasnovanog na sonarskim podacima.

3. Mjere za odre đivanje učinkovitosti algoritma planiranja prekrivanja velikih površina morskoga dna korištenjem autonomnoga plovila zasnovanog na sonarskim podacima.

Doktorski rad završava pregledom hipoteza i gore navedenih doprinosa te sažetkom najvažnijih točaka disertacije. Na temelju prezentiranog sadržaja ponovno se postavljaju te detaljnije razrađuju hipoteze i kao dokaz inovativnosti istraživanja nudi se popis publiciranih znanstvenih radova.

Ključne riječi: autonomna podvodna vozila, planiranje putanje prekrivanja, prekrivanje bazirano na senzorskim podacima, bočno-skenirajući sonar, procesiranje sonarskih slika, detekcija anomalija, detekcija meta, detekcija istaknutih objekata, modelski prediktivno upravljanje, podvodna lokalizacija

Contents

Chapter 1

Introduction

1.1 Motivation

Knowledge of the Earth and its evolving environment is proving increasingly crucial. Scientific, economic, political, and social decisions all depend at some time or another on this knowledge, and we like to think that we know all there is to know about our planet. One may be justified in doing so today, in the 21st century, by looking back at those maps with white unexplored regions that were still prevalent at the beginning of the 20th century. Yet, in many respects, we know more about the solid surface of other planets than about our own Earth. Rovers driving on Mars for years on end, landers on far-away Titan, and now the international missions to the Moon cannot mask the fact that ocean bottom landscapes only a few kilometers from our shores are still completely unknown [1].

More than half of the world's population live within 100 km of the sea. Thirteen of the 15 largest cities in the world are now located on or near the coast. The effects of denser population and accelerating climate change include the disappearance of ecosystems, coastal erosion, over-fishing, marine pollution, and higher vulnerability to marine disasters such as tsunami or volcanic activity. But the oceans cover more than two-thirds of the Earth's surface, and are not accessible to direct observation. It is only in the last 20 to 30 years that technological advances have allowed us to discover and map the Earth's seafloor, mostly through acoustic remote sensing [1].

Marine monitoring and exploration missions often include side-scan sonar based seafloor mapping. This includes the exploration of the biosphere, exploration of underwater archaeological sites, marine safety, and many other applications. Currently, side-scan sonar missions are executed either by a tethered towfish equipped with a side-scan sonar or by using remotely operated vehicles (ROVs) or autonomous underwater vehicles (AUVs). Deploying a tethered towfish from a boat requires hiring a

boat and its crew: a towfish operator, a side-scan operator, and of course experts in the scientific field for which the survey mission is being conducted.

Towfish missions are executed mostly in three phases. The first phase is a general survey mission at higher altitudes, whose goal is to identify possibly informative contents of the region being covered. In this phase, the boat tows a towfish in a lawnmower (LM) pattern to cover as much area as possible, without or with a low percent of overlapping side-scan sonar swaths. The goal of the second phase is to get some more detailed low altitude side-scan sonar images of some potentially interesting areas which are manually tagged within the first phase. This goal is achieved by using side-scan sonar's swaths overlaid to sonify interesting objects from more than one side. This way much more morphological information about the interesting objects can be extracted. The third and final phase includes deploying an ROV and/or an AUV at the several carefully selected locations of interest, and observing these locations usually with a high-resolution camera.

Using an AUV to perform side-scan sonar mapping during a mission that is a crossover between the first two phases of towfish survey can greatly simplify the whole mapping process, make it potentially significantly faster, cheaper in the long term, and safer.

1.2 Problem definition

The problem that is addressed in this thesis can be defined as: In order to cover a given unknown area of interest of dimensions $A \times B [m]$ with a given side-scan sonar at a constant surge speed u, and a constant depth/altitude h_{ref}, generate an initial Lawnmower (LM) coverage pattern of length L and width $W = 2w_{sss}$. During the mission adapt the coverage path depending on the detection of mission-specific interesting objects in sonar data. Thus complete coverage hear means getting the data from all informative objects (by sonifying them from both sides) as in case of the overlap-all-sonar-swaths LM pattern, while not necessarily sonifying uninteresting parts of the area from both sides. This way, assuming a large-scale area of approximately flat seafloor and a small percentage of that area covered by interesting objects, coverage path (or mission duration) can be significantly reduced without information loss about the detected interesting objects.

1.3   Thesis Contributions and Overview

The research goal of this thesis is to develop an online side-scan sonar data-driven coverage path planning (CPP) algorithm for monitoring and surveying large-scale (over $1km^2$) seafloor regions by an AUV, which would be much more convenient and less costly to deploy than a towfish. The main objective of the initial phase of this research is to analyze path planning algorithms and coverage path planning algorithms used for problems similar to the problem defined above, with special attention to their applications in marine robotics and side-scan sonar imaging.

The stated hypotheses are as follows:

1. An online dynamical programming-based coverage path planning algorithm can significantly reduce mission duration of side-scan sonar survey missions in a given large-scale area of the sea floor w.r.t. the mission duration of the usual overlap-all-sonar-swaths lawnmower pattern coverage plan.

2. An online heuristic decision making-based coverage path planning algorithm can significantly reduce the duration of survey missions, as well as mission (re)planning time w.r.t. the dynamical programming-based coverage planning algorithm, without loss in the quality of the final side-scan sonar imaging result.

3. It is possible to design a metric which would quantify performance of the above described survey mission objective. This metric can then be used to compare different coverage planning algorithms in terms of performance and robustness.

4. A reference scenario can be designed in order to validate the performance of the developed coverage path planning algorithms in a realistic simulation environment and in terms of the proposed performance metric.

One dynamical programming-based and three decision making-based coverage path (re)planning algorithms are proposed in this thesis. Their performance was thoroughly examined in a grid-like coverage area cost map simulation over a whole range of coverage mission parameters and randomly generated seafloor configurations to gain statistical metrics of their behavior. Also, their upper and lower performance bounds are mathematically modelled w.r.t. some mission parameters, and were validated by the thousands of simulation runs. Moreover, an existing realistic 3D simulation environment was extended by the above-mentioned coverage path planners, mission controller, and sonar data processing modules to gain further insights into the interplay of coverage path planning, feasible path interpolation, vehicle dynamics and control.

Therefore, scientific contributions of this thesis are summarized as follows:

1. An online sonar data-based coverage path planning algorithm for large-scale

seabed exploration missions using autonomous marine vehicles.

2. Simulation framework for implementing, testing, and evaluating performance of sonar data-based coverage planning algorithms.

3. Metrics for evaluating performance of sonar data-based coverage planning algorithms used in large-scale underwater survey missions.

The thesis is organized as follows: an introduction to the theory of navigation, guidance and control subsystems in autonomous underwater vehicles with a focus on model predictive control is given in the second chapter ("2. Navigation, Guidance and Control of Marine Vehicles"). The third chapter ("3. Sonar Data-based Coverage Path Planning Algorithms") provides a description of the aforementioned coverage algorithm based on sonar data. Their performance was compared with a classic model of a lawnmower in terms of mission duration/total length of coverage path, total (re)planning time, and statistical indicators of algorithm behavior for given mission parameters. The fourth chapter ("4. Object detection in side-scan sonar imagery") provides an overview of anomaly detection methods, i.e. generically defined interesting objects on the seabed that differ in any way from the environment, which is mostly just a flat sandy bottom. The fifth chapter ("5. Simulation environment for coverage planning algorithms validation") describes the developed modules integrated into the existing realistic 3D simulator of underwater autonomous vehicles in order to be able to validate the simulation results based on the coverage price matrix maps, and to analyze the mutual influences of the coverage planning modules, generation of feasible trajectories, vehicle control and processing of sonar data. Autonomous vehicles that were used for experimental validation of some of the aforementioned subsystems are described in the sixth chapter ("6. Systems"). The seventh chapter ("7. Case studies") presents five experimental studies related to the control of marine vehicles in order to follow the linear path, localization, recording of irregularly shaped areas, recording of underwater archaeological sites with multiple types of vehicles and integration of diverse data, and validation of the performance of sonar processing algorithms imagery with hardware in loop. The doctoral thesis ends with the elaboration of the hypotheses and contributions presented in the content of the doctoral thesis.

Chapter 2

Navigation, Guidance and Control of Marine Vehicles

2.1   Introduction

The navigation, guidance and control (NGC) system is responsible for directing the actuating forces and stabilizing the vehicle along the desired path. The NGC system, an example of which is given in Fig.2.1, has to define the trajectory in real time (shown as the blue "Path following controller" block in Fig.2.1) to reach the specified target and steer the vehicle along the desired path (shown as the green "Low-level speed controller" block in Fig.2.1). The guidance system generates the trajectory to achieve the target and desired vehicle steering command to realize the trajectory in real time. The vehicle control system, receives the steering commands from the guidance system and steers the vehicle to follow the desired attitude in the presence of all disturbances [2,3]. Section2.2is dedicated to introducing and testing the use of MPC for underactuated marine vehicles. The navigation system estimates the instantaneous state of the vehicle based on its model, current system output measurements, and previously estimated state and is represented as the red block "System states estimator" in Fig.2.1. A case study of AUV navigation improvement by sensor fusion, published by the author of this thesis in [4] is given in Section7.3.

Navigation, guidance and control of marine vehicles is a challenging task, mainly due to coupling between degrees of freedom (DoFs), hydrodynamic effects and highly unpredictable environmental influences. Sea operation and deployment cost for different types of autonomous marine vehicles is high, and risk of vehicle loss during untethered operation of Autonomous Underwater Vehicles (AUVs) is high. As a consequence, good mathematical models are needed to properly test and simulate NGC systems beforehand. Sensor and external disturbance modelling has to correctly ap-
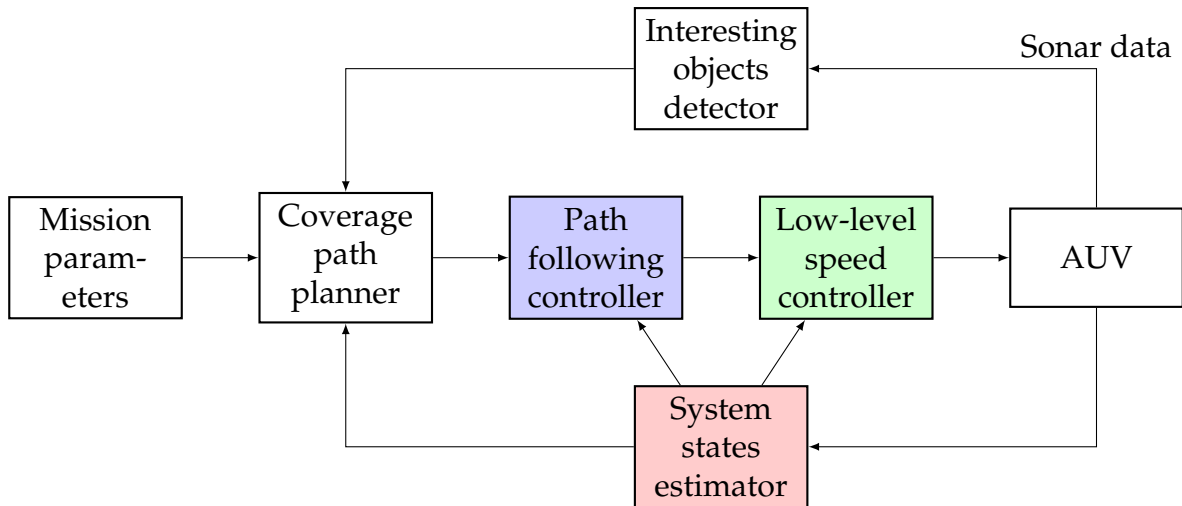
Figure 2.1: Block scheme of the entire AUV system. NGC subsystem is marked by colored blocks.

proximate the harsh and stochastic nature of underwater environments [5]. Therefore, before NGC development, the vehicle, from high-level kinematics down to thruster mapping, is mathematically modelled providing basis for navigation and controller tuning [3].

2.2 Model predictive motion planning

In this section a model predictive control (MPC) method is introduced as a mid-level yaw rate controller for AUV exploratory missions at constant depth with a predefined constant surge speed. MPC has advantage over the classical control methods because it does not generate its current control signal(s) based solely on the previous value of the state error vector w.r.t. the steady-state or reference state value(s). Instead, it also takes into account the mathematical model of the system and based on it and the constraints which are imposed on system states, it predicts the behavior of the system on some prediction horizon of an arbitrary duration. This duration is of course limited by the sampling period and computational power of vehicle's onboard computer. When the optimal control problem is solved on this prediction horizon, and the best solution is chosen w.r.t. some cost function, only a portion of the optimized control signals is applied to the system. Of course for a real-time control of a vehicle with relatively fast dynamics this open-loop control is set to the duration of one sampling period, after which the system states are again estimated and the negative feedback is closed. Parts of this section were previously published by the author in [6–9]. The block scheme showing the overall view of the system with MPC controller as the "Path following controller" (in red) is given in Fig.2.2.
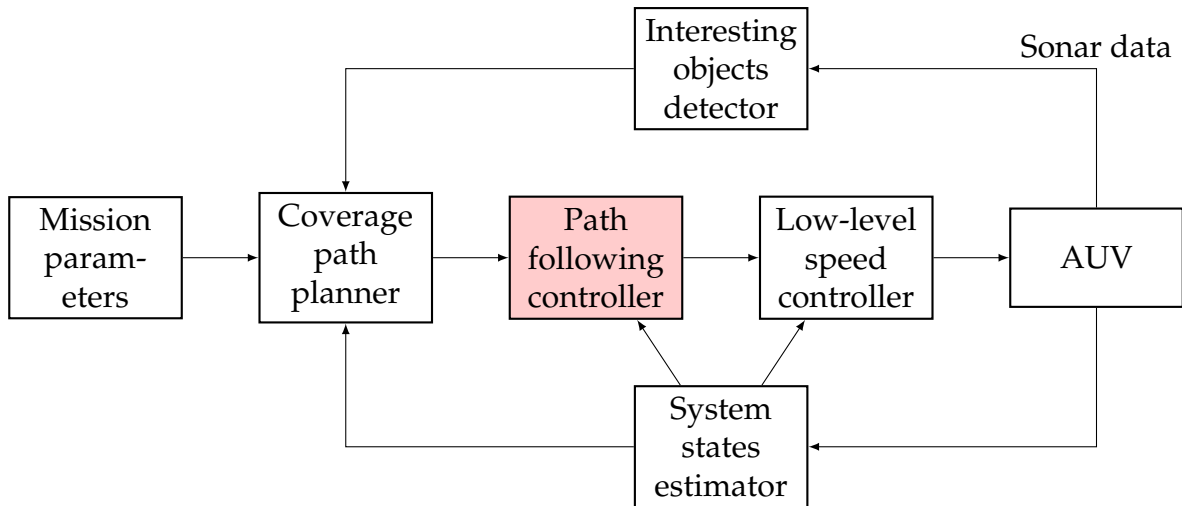
Figure 2.2: Block scheme of the entire AUV system.

The rest of this section is organized as follows: a short introduction of model predictive control (MPC) general concepts and framework is given in Section 2.2.1. An overview of model predictive motion planning methods for ground vehicles as well as marine vehicles is given in Section 2.2.2. Conceptual discussion about the design of an MPC framework which takes into account the exteroceptive sensor(s) data into the process of path planning and path following is given in Section 2.2.3. Kinematic model of line following with constant disturbance that was used for MPC is presented in Section 2.2.4. Section 2.2.5 describes a linear MPC scheme for path following having in mind that its input path is actually an output from a path planning module, more of which is given in Chapter 3. Tuning the MPC controller is a very important step in order to optimize its control but also execution time performance. Extensive simulations with MPC parameters variation and the metric used to evaluate MPC controller performance are given in Section 2.2.6. Experimental results which validated the simulation results are presented in Section 7.2.

2.2.1 MPC: A short introduction

Lets assume a following situation: some person is driving a car. The person knows some part of the road (path) ahead on a finite horizon. Also, the person knows the behavior of the car (mental model of the car), and thus decides which control actions to take (accelerator, brakes, and steering) in order to follow the desired path on the road. It is important to add that the driver in this situation makes decisions about controls by looking both backwards in the mirrors (past errors), but also buy knowing the desired path which needs to be followed and predicting the behavior of the car in advance. The controls are applied at each time instant, while the whole process of
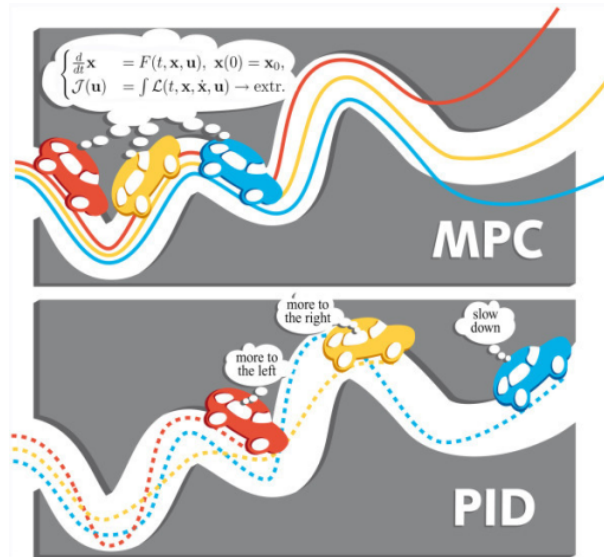
Figure 2.3: Comparison of the way MPC and PID work [11].

prediction and control generation are repeated. This car analogy from [10] depicts the main principles of MPC. In the classical control schemes, such as PIDs, the decisions about which control action to take are based only on looking backwards in the mirrors (past errors), while in MPC schemes they are based on looking both backwards but also forward, thus predicting the behavior of the car. This comparison of MPC and Proportional-Integral-Derivative (PID) control is graphically represented in Fig.2.3.

Model predictive control (MPC) can be described as repeated solving of the Optimal Control Problem (OCP) on a finite time horizon T_p in open loop, while taking into account system dynamics and constraints imposed on the states and the controls . Optimization is executed in each sampling time instant $t_i = iT_s$, $i = 0, 1, 2, \ldots$, where T_s is the sampling period. Estimated states at each time instant t_i are used as the initial conditions of the OCP. Even though the solution of the OCP are controls in the open loop, an implicit feedback is made with these initial conditions of the OCP. This way, the MPC is made more robust w.r.t. the measurement and model noise, but also disturbances. When the optimized controls are acquired, only a part of the duration T_c is applied to the system, and in the next sampling instant t_{i+1} the whole procedure is repeated. T_c is the so-called control horizon, and it is most often set as $T_c = T_s$. MPC can be, generally speaking, applied for nonlinear and linear, continuous and discrete systems, with hard and soft constraints [10].

The underlying concepts behind MPC are not novel. They include Hamilton-Jacobi-Bellman theory of dynamic programming [12], the maximum principle of optimality, and Kalman's observation that optimality does not imply stability [13]. MPC has been used since the 1970's in the areas of oil refining, petrochemicals and chemicals, but

also pulp and paper, food processing, aerospace and automotive industries, gas, utility, furnaces, mining and metallurgy [10,14]. The clock frequencies of processors are constantly increasing, thus enabling faster computation of the OCP even for complex systems. Therefore, MPC has long been used in the fields of industrial, field, aerospace, and underwater robotics.

The general mathematical formulation of the MPC problem goes as follows: Let us consider stabilizing a time-invariant (non)linear system described by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(0) = \mathbf{x}_0 \tag{2.1}$$

w.r.t. the state and control constraints:

$$\mathbf{u}(t) \in \mathscr{U}, \qquad \forall t \geq 0 \tag{2.2}$$

$$\mathbf{x}(t) \in \mathscr{X}, \qquad \forall t \geq 0 \tag{2.3}$$

The solution of this problem is a control signal computed as a solution of the optimal control problem

$$\min_{\overline{\mathbf{u}}(\cdot)} J(\mathbf{x}(t_i), \overline{\mathbf{u}}(\cdot)) \tag{2.4}$$

w.r.t. the constraints

$$\dot{\overline{\mathbf{x}}} = \mathbf{f}(\overline{\mathbf{x}}(\tau), \overline{\mathbf{u}}(\tau)), \qquad \overline{\mathbf{x}}(t_i) = \mathbf{x}(t_i), \tag{2.5}$$

$$\overline{\mathbf{u}}(\tau) \in \mathscr{U}, \overline{\mathbf{x}}(\tau) \in \mathscr{X}, \qquad \tau \in [t_i, t_i + T_p], \tag{2.6}$$

$$\overline{\mathbf{x}}(t_i + T_p) \in \mathscr{E} \tag{2.7}$$

where $\mathscr{E}$ is called the terminal region, and (2.7) is called the terminal region constraint, which enforces system stability. The bar above the state and control vector symbols in (2.4)-(2.7) denotes prediction variables internal to the controller itself. This distinction between the real system's and its model's variables is necessary because their values will not be the same in the general case.

The cost function J which is being minimized on the prediction horizon $T_p \geqslant T_s \geq 0$ is usually given as

$$J(\mathbf{x}(t_i), \overline{\mathbf{u}}(\cdot)) := \int_{t_i}^{t_i + T_p} F(\overline{\mathbf{x}}(\tau), \overline{\mathbf{u}}(\tau)) \, \mathrm{d}\tau + E(\overline{\mathbf{x}}(t_i + T_p)), \tag{2.8}$$

where function F is called the stage cost or Lagrange term, and function E is often called the terminal cost or Mayer term. The terminal cost E, i.e. the terminal region constraint (2.7), does not need to be included in the cost function. The mathematical formulation given here lacks mathematical rigour. It is given to describe the general

concepts of the MPC without going into details.

Since the OCP described by (2.4)-(2.8) has to be solved on-line, this implies that the finite horizons have to be used. The shorter the prediction horizon is, the less complex and time consuming it is to solve the given OCP. However, if the finite prediction horizon is used, closed-loop control and system trajectories will differ from the predicted trajectories in open-loop, even if there is no model noise or measurement noise present [15]. Also, infinitely repeated finite horizon optimization in the sense of the receding horizon does not lead to the optimal solution on the infinite horizon [16]. A comprehensive overview of the MPC's stability analysis is given in [17].

2.2.2 MPC based motion planning

MPC-based motion planning on rough terrains

Planning mobile robot's motion along the path from some starting location to the desired goal location, while at the same time minimizing some cost e.g. roughness of the traversed terrain, can be done by using some of the classical path planning algorithms (Dijkstra, A* for known terrains, or D*, Field D*, D* Lite for unknown of partially known terrains, etc.). The resulting path, going from the start to the goal, usually has sharp turning points, since the above mentioned algorithms are grid-based. This way of generating the path does not consider the dynamics of the vehicle, its kinematic and/or kinetic model. Smoothing of these paths can lead to the loss of optimality, and even worse, the vehicle may hit an obstacle, roll-over, or slip, while following the smoothed path.

The success of the mission, i.e. vehicle traversing the rough terrain from the starting to the goal location, can be guaranteed if the path planner is designed in such a way that it gives the trajectory of the vehicle w.r.t the vehicle's model and state and/or control constraints (in the sense of avoiding obstacles, preventing slipping, and rolling-over), and the controls needed to execute that trajectory. MPC seems like an intuitive choice for tackling this problem.

One such MPC-based planner for mobile robots on known large-scale rough terrains is presented in [18]. It uses the interpolated roughness values, i.e. roughness function as the Lagrange term in the OCP, for locally planning the smoothest path. Also, the interpolation of the roughness-to-go map is used as the Mayer term in the formulated OCP. Roughness-to-go map represents a global planner along the smoothest path from the edge of the prediction horizon to the goal position. Roughness-to-go map has been calculated by using either the optimal Dijkstra algorithm, or the suboptimal DRoughness based Navigational Function (RbNF) algorithm. Since RbNF al-

gorithm cannot cope with arbitrarily large obstacles in order to give a near-optimal cost-to-go map, it was further improved in [19]. Wavefront Roughness based Navigational Function (WRbNF) algorithm presented in [19] is a near optimal cost-to-go map and it can also be used for large scale rough terrains with an arbitrary number and size of obstacles. The algorithm has an inherent parallel structure and the code can be parallelized, so it can be significantly faster than the Dijkstra algorithm when an adequate amount of computer resources is used. In [9], WRbNF algorithm was used as a terminal cost term in the MPC-based path planner for mobile robots on rough terrains.

Another solution, given in [20], is to use D* algorithm as a global planner for path planning on the unknown or partially known rough terrains. The D* cost is then used as the terminal cost in the MPC, while using the interpolated roughness values for local planning.

MPC-based motion planning in marine robotics

MPC has started being used in marine robotics field approximately from the 2000's onward. This is most probably due to the speed up of the computational power of computers, as well as vehicle on-board microcontrollers.

MPC scheme for path planning in a dynamically changeable environment with obstacles and sea currents is presented in [21]. Map of the environment is built using forward-looking sonar data, and D^* algorithm is used to generate waypoints, interpolated by using B-splines, for MPC to follow. Sampling based model predictive control (SBMPC) scheme with input sampling and obstacle avoidance is presented in [22]. MPC can be also used for safety motivated path planning, e.g. for ships at the open sea with high waves, in order to minimize the possibility of capsizing due to roll and pitch oscillations [23]. Strong currents have a great impact on energy consumption of the AUVs. MPC approach using A* algorithm for computing the path along the weakest current flows in the ocean in presented in [24]. A comprehensive overview of the MPC framework for path planning schemes is given in [25].

Even with the use of path planning algorithms, there remains the issue of tracking the path which was computed. This can pose a problem, especially in the case of unfeasible sharp turns between consecutive waypoints along the path. Path tracking is solved by MPC in a number of approaches. MPC scheme with a Shrinking horizon model predictive control (SHMPC) is presented in [26]. This approach shrinks the prediction horizon as the vehicle approaches the goal position, in order to relieve the computation burden of optimization solver. To reduce the posibility of capsizing in ship maneuvering, it is important to constrain roll and rudder turning rate, together with canceling the disturbances caused by high waves. MPC has been used to tackle

this problem in [27,28]. Piece-wise linear paths can be tracked very efficiently using Line-of-Sight (LOS) approach, which was integrated into MPC framework in [29,30].

2.2.3 Informational gain guided MPC motion planning: A transition from land to sea

In this subsection, we discuss two possible frameworks of the informational gain guided MPC based motion planning for the AUV during sonar scanning missions.

Characteristic waypoint following

The first approach to informational gain guided MPC based motion planning for the AUV is pretty straightforward, see Fig.2.4. It uses gathered sonar data to compute informational gain values of the scanned sea floor, and uses a D*-like (near-) optimal global path planner. This planner at its output gives a series of waypoints in the grid map for the AUV to follow. Since these paths often contain many sharp turns which the AUV cannot perform, it is necessary to smooth out the computed path, e.g. by using B-spline basis functions. The problem which remains with this smoothing process is that, in the presence of obstacles, the smoothed path can collide with them, even though the original path's waypoints perfectly avoid the obstacles.

After a global or local parametrization of the path with a smooth continuous function, MPC can be used for path following, as some sort of a local planner. Of course, MPC with AUV's kinematic model can be used for high-level control, giving the reference velocities to the low-level controllers, which in turn give forces and torques at their output. Another option is to include both kinematic and dynamic models into MPC, so that it directly optimizes forces and torques. The decision on which MPC model to use depends mainly on the fact whether the optimization execution times are still real-time.

Informational gain guided MPC

Another approach would be to integrate the cost-to-go values from the global planner directly into the MPC optimization framework, see Fig.2.5. This can be done as in [9, 20], to use the global planner values as the terminal cost term E in (2.8). It is also possible to extend the local MPC planner F with the locally interpolated informational gain values. This way, the vehicle would move along the path which is maximizing the informational gain, both locally and globally.

Adding the terminal cost in a form of the interpolated values of the cost-to-go map, and/or adding the interpolated local informational gain in the optimization criterion
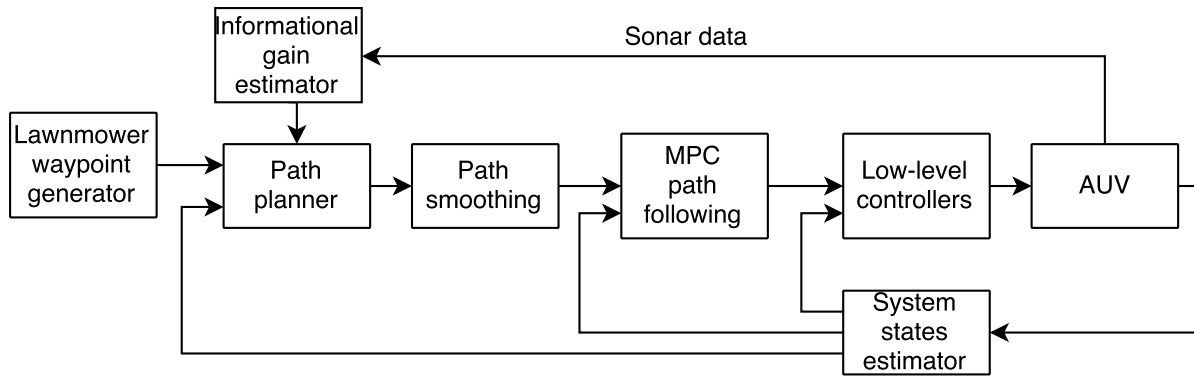
Figure 2.4: Block diagram of the first proposed approach: Global path planner for computing the discrete path in the grid based map, which is then smoothed, and tracked with MPC path following control algorithm as a local planner.
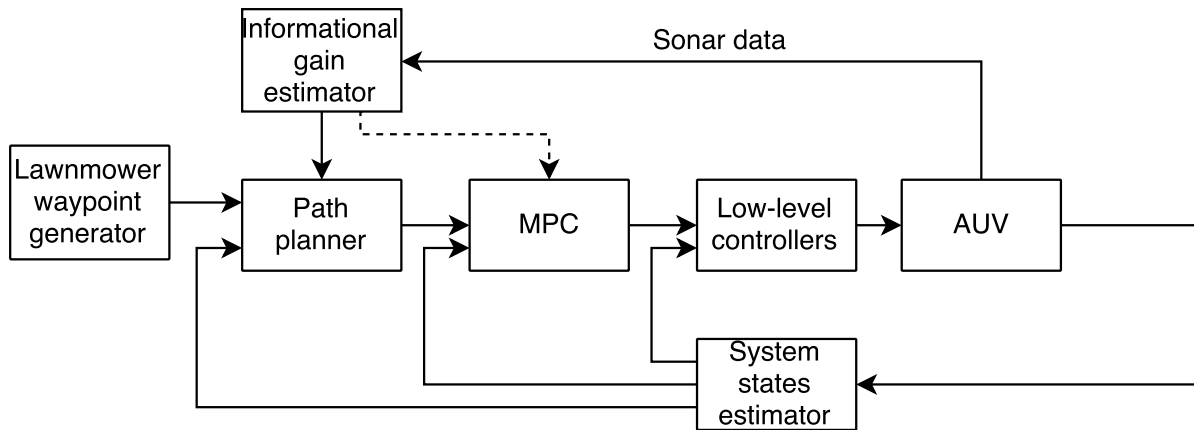

Figure 2.5: Block diagram of the second proposed approach: Global path planner for computing the terminal cost in the MPC framework. Possible augmentation of the local MPC planner with the roughness values on the prediction horizon (dashed line).

of the OCP can cause the optimization problem to become more complex to solve. Of course, there should be a trade-off between the complexity which is imposed on the optimization solver, and the vehicle's performance improvement.

2.2.4 Kinematic model of line following with constant disturbance

Let us assume that we need to use some underactuated marine vehicle for a sea floor sonar scanning mission. Also, let us assume that for accomplishing that mission, the vehicle should move only in a constant depth plane. In this context, we define underactuatedness of the vehicle as its inability to directly control pitch, roll, and sway. Heave motion is neglected since we are assuming that the vehicle is already at the desired depth. Moving in the marine environments necessarily means dealing with different kinds of disturbances, i.e. wind and waves for missions on the surface, sea

current for underwater missions. These disturbances can also represent unmodeled system dynamics. Let us furthermore assume that these disturbances are constant, but not observable, and denote them as ζ_d. Since we are assuming a sonar scanning mission, it is preferable for the vehicle to maintain a constant surge speed w.r.t. water $U_r > 0, U_r > \nu, U_r = const.$, while the change in heading is controlled by yaw rate r.

This leaves us with a task to control yaw rate in such a way that the vehicle follows the straight lines of the desired lawnmower pattern, i.e. minimizing the distance to the current line (cross-track error) d. Also, heading error $\beta = \psi - \psi_d$, i.e. the difference between the vehicle's heading ψ, and current line's heading ψ_d, should be minimized.

With all the above assumptions taken into account, kinematic model of line following in a horizontal plane is given as

$$\dot{d} = U_r \sin \beta + \nu \simeq U_r \beta + \zeta_d \tag{2.9}$$

$$\dot{\beta} = r \tag{2.10}$$

$$\dot{d_{int}} = d \tag{2.11}$$

where the symbol $\simeq$ denotes linear approximation for small values of β [31].

The linearized kinematic model of line following given by (2.9) and (2.10), is thus extended with an additional state given by (2.11), which represents the integral of the cross-track error.

Furthermore, vehicle's position and orientation $[x\,y\,\psi]$ in the earth-fixed frame $\langle e \rangle$ are expressed as

$$\dot{x} = U_r \cos \psi + \zeta_x \tag{2.12}$$

$$\dot{y} = U_r \sin \psi + \zeta_y \tag{2.13}$$

$$\dot{\psi} = r \tag{2.14}$$

where ζ_x, ζ_y are x and y components of the current speed, respectively [31].

2.2.5 Linear model predictive control framework for line following

In order to minimize the computation complexity, and be able to implement our control framework on a real marine vehicle, in our approach we used linear MPC as a high level yaw rate controller. Linearized kinematic model of line following with constant disturbance from Eqs. (2.9)-(2.14) was used Low level PID controllers are delegated to control constant surge speed throughout the whole mission, but also to track the reference yaw rate set by the high level MPC controller.

This model extension is a common practice in robust MPC schemes with distur-

bance rejection, as in [32], which enables stabilization of the system around the setpoint even in the presence of an external disturbance.

This integral state should not be confused with the direct integration block which is added to the system in the PID control schemes for line following in order to reject the disturbance, as in [31]. State d_{int} is merely implicitly, through its presence in the cost function, causing MPC controller to take into account the effects of the disturbance, thus optimizing the control which rejects it.

In order to solve control optimization in a MPC fashion, we used Toolkit for Automatic Control and Dynamic Optimization (ACADO) [33], as the control optimization tool. Optimization was even more sped up by the use of RealTimeAlgorithm class, which enabled us to make MPC controller ready to be used in real-time. MPC framework which has been used is given here. Cost function J is expressed as

$$J = \int_{t_i}^{t_i+T_p} \left(K_d d^2(\tau) + K_\beta \beta^2(\tau) + K_{d_{int}} d_{int}^2(\tau) \right) d\tau \qquad (2.15)$$

subject to

$$-\pi \leq \beta(\tau) \leq \pi, \qquad\qquad \forall \tau \in [t_i, t_i + T_p] \qquad (2.16)$$

$$-20°/s \leq r(\tau) \leq 20°/s, \qquad\qquad \forall \tau \in [t_i, t_i + T_p] \qquad (2.17)$$

where $t_i = kT_s, k \in \mathbb{N}_0$ is initial time of the prediction horizon which lasts for $T_p[s]$, with sampling time $T_s = 125ms$. Another MPC design parameter in ACADO was N_{steps} parameter. This parameter implicitly sets the duration of the control horizon, since $T_c = N_{steps}T_s[s]$ holds. Terms N_{steps} and T_c will be used interchangeably from this point forward.

2.2.6   Simulation results

Tuning of MPC controller is often done ad hoc, choosing its parameters which give just good enough results. Our idea was to vary MPC controller's parameters (T_p, N_{steps}, K_d, K_β, and $K_{d_{int}}$) for the particular model and environment setting in some relatively wide, but sensible range, simulate the system in the closed-loop, and asses the quality of the system performance numerically. Problem space for tuning all the parameters at the same time was too big, so we divided it into two stages of parameter variation. In the first simulation stage, T_p and N_{steps} were varied, and in the second stage K_d, K_β, and $K_{d_{int}}$ were varied.

It is important to note here that the set value of the surge speed for the vehicle in the simulations was $U_r = 0.5m/s$, and that the disturbance was simulated as $\nu = \nu_x =$

$0.2 m/s$, $\nu_y = 0$. Steady-state ϵ-zone is defined as $\epsilon = 0.1m$.

Indices for line following performance evaluation

Since in the parameter variation simulations many responses get generated, it is of our interest to grade these responses in some objective way. This was done using the methodology proposed in [34]. It enabled us to analyze and quantify performance of different parts of the vehicle's path towards convergence to the desired line, i.e. turn phase, approach phase, and settling and steady-state phase. Graphical explanation of all the parameters in all three phases is given in Fig.2.6.

As defined in [34], line following performance during the turn phase is parametrized by $H_{||}[m]$, $A_1[m^2]$, and $H_\perp[m]$, see Fig.2.6.

The second phase is the so-called path approach phase, and it is parametrized by $A_2[m^2]$, $\overline{\chi}[m/s]$, and $\overline{\chi}_{max}[m/s]$, see Fig.2.6. Here $\overline{\chi}[m/s]$ denotes the mean of cross-track error rate normalized by the duration of the approach phase, while $\overline{\chi}_{max}[m/s]$ is the maximum value of cross-track error rate during the approach phase.

The third and the last phase is the settling and steady-state phase, in which cross-track error is within some ϵ-zone around the followed path. This phase is parametrized by four parameters, namely $H_2[m]$, $A_3^*[m]$, $t_s[s]$, $\overline{\Delta r}[\deg/s]$, see Fig.2.6. Settling time t_s has been added as an additional analysis parameter. Also, instead of computing the rudder stress R as in [34], here we computed yaw rate stress index $\overline{\Delta r}$, and thus implicitly the energy consumption.

In addition to using the quantitative analysis of line following from [34], in this approach all of the indices quantifying the response quality were scaled relative to the minimum/maximum value of that index throughout all simulations. The scaling was done so that we could easily compare system response qualities from our simulations, and make some generalized weighted cumulative performance function denoted as Σ. The value of this performance measure can tell us for which MPC parameter n-tuple we get the best control performance.

Scaling by the minimum value of the performance index was done due to the fact that smaller values of $H_{||}, A_1, H_\perp, A_2, H_2, A_3^*, t_s, \overline{\Delta r}$ mean better system performance. Conversely, scaling by the maximum value was done due to the fact that greater values of performance indices $\overline{\chi}$ and $\overline{\chi}_{max}$, and also of the overall performance measure Σ, mean better system performance. If we generalize parameter p as $p \in \{H_{||}, A_1, H_\perp, A_2, \overline{\chi}, \overline{\chi}_{max}, H_2, A_3^*, t_s, \overline{\Delta r}, \Sigma\}$, then the scaling by the minimun value is given by

$$p_{ij,s} = \frac{min\{p_{1j}, p_{2j}, \cdots, p_{N_{sims}j}\}}{p_{ij}} 100[\%] \tag{2.18}$$
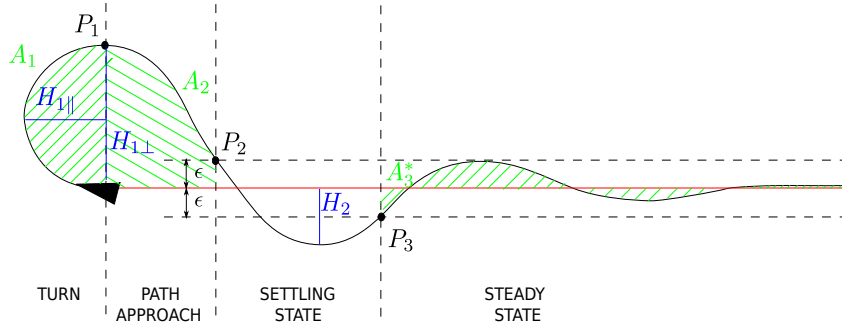
Figure 2.6: Performance indices of path following [34].

and scaling by the maximum value of the performance index is given by

$$p_{ij,s} = \frac{p_{ij}}{max\{p_{1j}, p_{2j}, \cdots, p_{N_{sims}j}\}} 100[\%] \tag{2.19}$$

where $i = \overline{1, N_{sims}}$ denotes the index of the simulation, $j = \overline{1, N_{params}}$ denotes the index of the performance index, N_{sims} denotes the overall number of conducted simulations, i.e. the number of specific parameters' combinations, and N_{params} denotes the number of performance indices evaluated. The overall performance score Σ_i of one set of varied MPC parameters is calculated as a weighted sum of all scaled performance indexes for i-th simulation, given by

$$\Sigma_i = \sum_{j=1}^{N_{params}} w_j p_{ij,scaled} \tag{2.20}$$

where w_j denotes the weight, i.e. the importance of j-th parameter. All weights were as $w_j = 1, \forall j = \overline{1, N_{params}}$. After this, all the performance scores are scaled by the maximum principle given by (2.19).

Choosing prediction and control horizon

The first stage of parameter variation procedure was to vary parameters T_p and N_{steps} (thus implicitly varying the control horizon T_c). Prediction horizon duration T_p has a very important role in the overall stability of the controlled system in closed-loop. Apart from varying T_p and N_{steps}, we also varied the initial distance of the vehicle w.r.t. the line to follow, in order to choose T_p such that the system stabilizes in both cases.

On the other hand, control horizon duration T_c can cause the system to destabilize if T_c is chosen too short. If T_c is chosen too long, it causes faster response in most cases, but also an additional control optimization burden. Large value of T_c can also cause the controls to change too rapidly, which can lead to shortened life of actuators in the long term.

The first set of simulations was conducted with varying $T_p \in \{1, 5, 10, 20, 30, 60\}[s]$ and $N_{steps} \in \{1, 5, 10, 15, 20\}$. The initial conditions of controller internal system model were $d_0 = 0, \beta_0 = \pi, d_{int_0} = 0$. For these experiments cost function parameters were set as $K_d = 1$, $K_\beta = 0.001$, and $K_{d_{int}} = 0.01$. For each T_p the best response with corresponding T_c has been shown in Fig.2.7a. The second set of initial conditions was $d_0 = 5, \beta_0 = \pi, d_{int_0} = 0$. The corresponding best system responses for each T_p are shown in Fig.2.7b.

It can be noted that longer T_p cause the response to have a larger overshoot (see also H_2 column of Table2.1). Also, shorter T_p caused the system to have a longer settling time, which is apparent in t_s column of Table2.1.

It is interesting to note that the vehicle's heading in the steady-state phase is not aligned with the followed line's heading, see Fig.2.7aand2.7b. This is due to the fact that the vehicle orients itself at an angle to compensate against the current at the assumed constant surge speed.

For the sake of brevity, we did not include the turn phase indices values here. We found that all three turn phase performance indices have the same values $H_\| = 1.46m$, $A_1 = 8.49m^2$, and $H_\perp = 1.17m$, invariant to the chosen stabilizing MPC parameters. The vehicle tries to turn as fast as possible in this phase, so these indices depend only on the surge speed and yaw rate bounds.

Table2.1shows the unscaled performance indices of the approach, and steady-state phase, but also the overall scaled score of each $T_p - N_{steps}$ combination. It can be noted that the score does not grow with the increase of T_p, but is some nonlinear function of T_p. This leads to the conclusion that longer prediction horizon does not always guarantee better MPC controller performance.

Also, it can be noted that as the fraction T_c/T_p increases, the vehicle approaches the line faster. This means that, as T_p gets shorter, the system cannot predict its future trajectories far in time, so as T_c takes up a bigger part of T_p, MPC generates more aggressive controls (see $\overline{\Delta r}$ column of Table2.1and Fig.2.7c) which in turn cause the vehicle to approach the line faster. Based on Table2.1, we chose $T_p = 30s$, and $N_{steps} = 10$.

Another thing which we were trying to asses empirically, was the estimation of the optimization duration in each sampling period. It is important to note here that the machine which was used for simulations had an Intel® Core™ i5-3210M processor with clock frequency of $2.5 - 3.1GHz$, and $6GB$ of Random-access memory (RAM). In Fig.2.7dit can be seen that the distributions of execution times of simulations for each pair (N_{steps}, T_p) have been repeated 10 times to gain some statistical credibility. It is evident that the complexity, i.e. execution time is dependent only on the number of
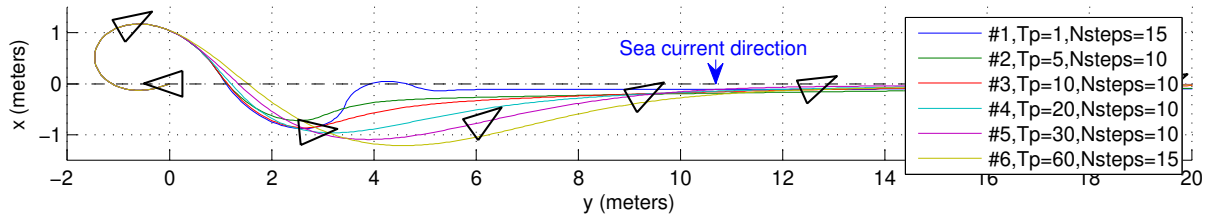
(a) Vehicle's path in a constant depth plane during line following maneuver . Reference line (dashed black line). Initial conditions $d_0 = 0[m]$, and $\beta_0 = \pi[rad]$. Heading of the vehicle (black triangles).
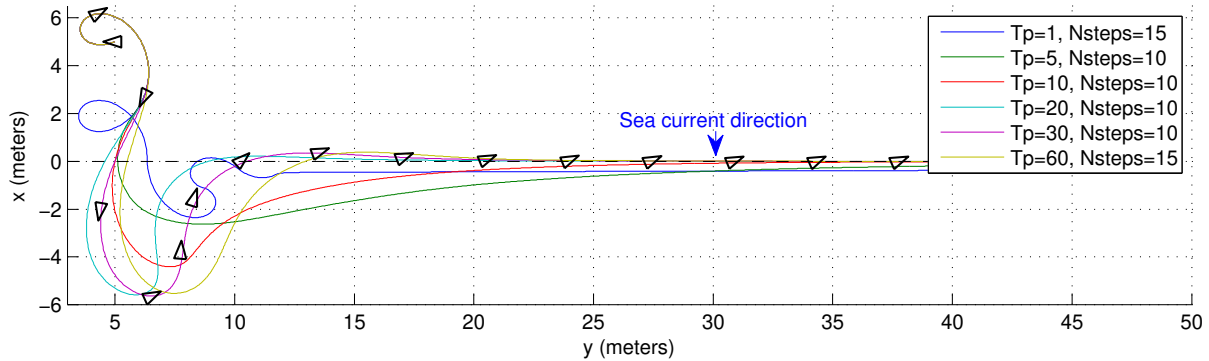
(b) Vehicle's path in a constant depth plane during line following maneuver . Reference line (dashed black line). Initial conditions $d_0 = 5[m]$, and $\beta_0 = \pi[rad]$. Heading of the vehicle (black triangles).
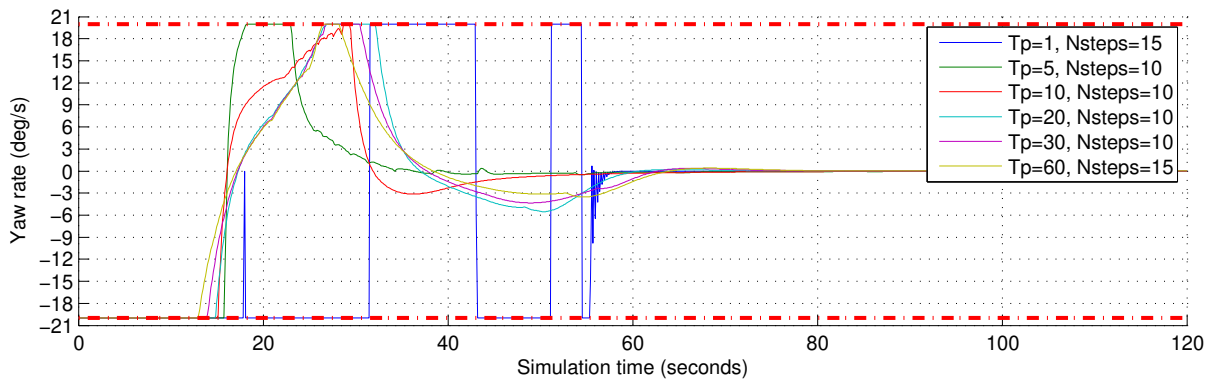
(c) Yaw rate for different Tp and T_c. Initial conditions $d_0 = 5m$, and $\beta_0 = \pi$. Constraints on yaw rate (red dashed line).
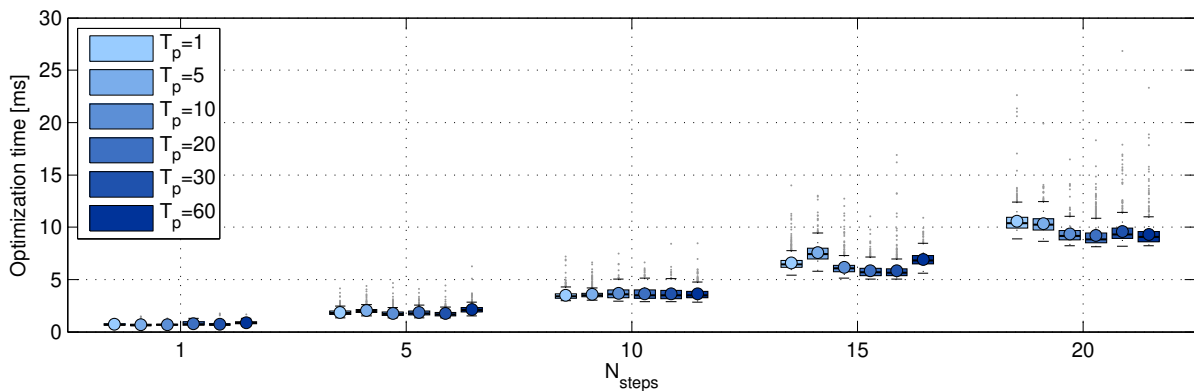
(d) Statistical analysis of control optimization execution times per one sampling period.

Figure 2.7: Simulation results for for various Tp and N_{steps} values.

Table 2.1: Prediction and control horizon variation. Quantitative analysis of response quality of the approach and steady-state phase, control stress, and final score. Initial conditions: $d_0 = 0$, $\beta_0 = \pi$, $d_{int_0} = 0$. For # referencing to T_p and N_{steps} values see Fig.2.7a legend.

#	A_2	$\overline{\chi}$	$\overline{\chi}_{max}$	H_2	A_3^*	t_s	$\overline{\Delta r}$	$\Sigma_{i,s}$
1	2.984	0.039	0.069	0.88	0.105	56.5	4.54	77.8
2	2.986	0.038	0.065	0.72	0.048	50.8	1.31	86.9
3	2.988	0.038	0.064	0.86	0.026	39.8	1.19	92.9
4	3.016	0.037	0.059	0.96	0.018	37.0	1.12	97.3
5	3.105	0.036	0.053	1.09	0.016	34.0	0.86	100
6	3.283	0.033	0.048	1.21	0.019	38.6	0.72	94.5

control steps being optimized at each MPC iteration For $N_{steps} = 10$, execution times for any tested T_p are under $5ms$, which is still far below the value of sampling time $T_s = 125ms$. Even with the transition to real system's lower performance on-board computer, this way we could make sure that the execution time of OCP solver will be real-time.

Choosing cost function parameters

After choosing prediction horizon T_p and control horizon T_c, the next step was to choose the parameters of cost function J given by (2.15). Again, this was done through parameter variation analysis for $K_d \in \{0.01, 0.1, 1, 2, 5\}$, $K_\beta \in \{0.001, 0.01, 0.1, 2, 5\}$, and $K_{d_{int}} \in \{0.001, 0.01, 0.1, 1, 2\}$. The best 6 resulting responses are chosen, and shown in Fig.2.8.

Table 2.2 shows the unscaled performance indices of approach and steady-state phase, but also the overall score of each $T_p - N_{steps}$ combination. Based on Table 2.2, we chose $K_d = 1.0$, $K_\beta = 5.0$, and $K_{d_{int}} = 0.001$. Indeed, the response with these parameters converges to the steady-state much faster than most of the other considered responses, see Fig.2.8. Also, its overshoot over the line is second smallest among all other responses taken into account.
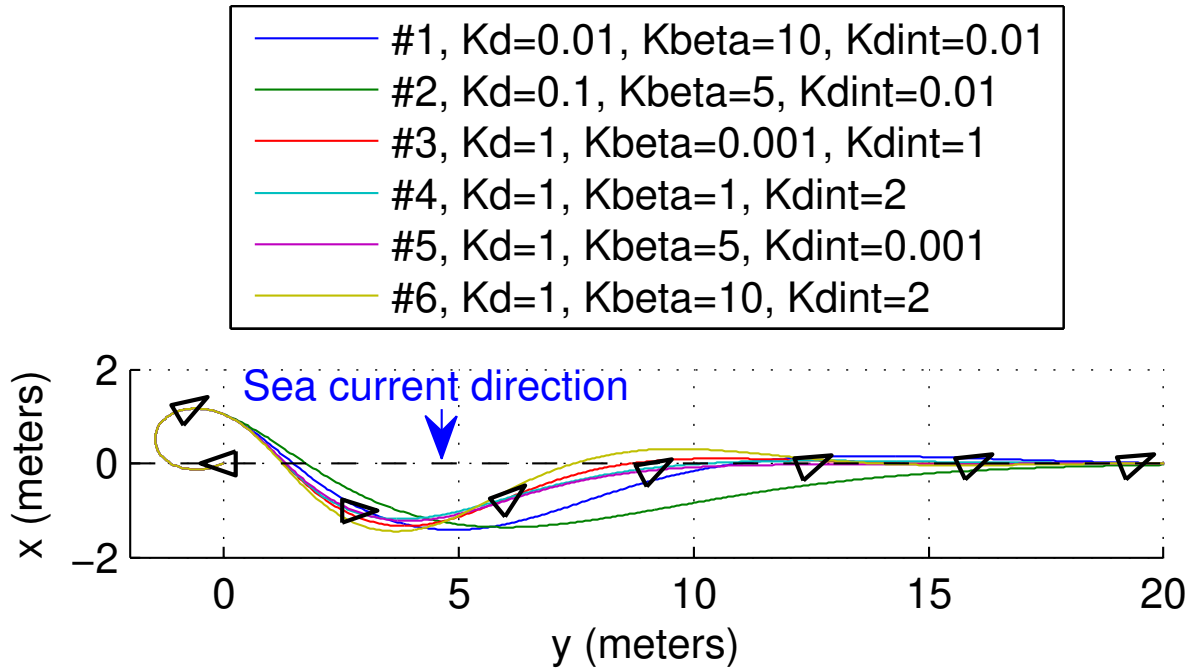
Figure 2.8: Cost function paratemers variation for $T_p = 30s$ and $N_{steps} = 10$. Initial conditions $d_0 = 0[m]$, and $\beta_0 = \pi[rad]$. Reference line (dashed black line). Heading of the vehicle (black triangles).

Table 2.2: Cost function parameters variation. Quantitative analysis of response quality of the approach and steady-state phase, control stress, and final score. Initial conditions: $d_0 = 0$, $\beta_0 = \pi$, $d_{int_0} = 0$. For # referencing to K_d, K_β, and $K_{d_{int}}$ value see Fig.2.8 legend.

#	A_2	$\overline{\chi}$	$\overline{\chi}_{max}$	H_2	A_3^*	t_s	$\overline{\Delta r}$	$\Sigma_{i,s}$
1	3.21	0.0334	0.048	1.42	0.01	46.0	0.7	94.5
2	3.47	0.0302	0.041	1.36	0.0094	47.3	0.6	94.9
3	3.07	0.0362	0.056	1.33	0.009	34.8	0.9	99.8
4	3.08	0.0358	0.054	1.19	0.0133	30.1	0.9	95.5
5	3.09	0.0355	0.054	1.22	0.0097	31.5	0.8	100
6	3.04	0.0365	0.058	1.45	0.0089	38.6	1.1	98.1

2.3   Conclusion

This chapter brought a brief introduction about navigation, guidance and control systems of autonomous marine vehicles and their challenges. MPC has been shown to have good performance when used for motion planning of the underactuated autonomous marine vehicle. Its main advantage as a motion planning method is that it generates kinematically (or even dynamically, depending on the model being used) feasible paths, for which it optimizes control signal(s) in such a way that the given constraints on system states and controls are met. This method takes into account the model, cost function, and constraints during the process of control signal optimization, as opposed to other methods such as PID and Lyapunov Control Function (LCF) control, which do not deal with the constraints explicitly, and satisfy them by eventually saturating the generated control signal. This *a posteriori* saturation can lead to significant system performance deterioration and even system instability in some cases.

A modular MPC framework for general use has been developed using open-source software ACADO toolkit, and it has been integrated with Robot Operating System (ROS). Depending on the control problem, the system model, constraints, cost functions and algorithm parameters can be easily changed. Real-time MPC framework implementation has been achieved for the line following problem which has been addressed in this section. Simulation and experimental results show good performance of MPC controller when compared to PID controller. The main advantage of MPC over PID control in this perspective is its possibility to be used as a general control optimization framework.

Next chapter brings introduction to coverage path planning algorithms in general, but also specifically for autonomous marine vehicles and presents the developed coverage path planning algorithms and their performance in simulation environment.

Chapter 3

Sonar Data-based Coverage Path Planning Algorithms

3.1 Introduction

Marine monitoring and exploration missions often include side-scan sonar based seafloor mapping. This includes the exploration of the biosphere, exploration of underwater archaeological sites, marine safety, and many other applications. Currently, side-scan sonar missions are executed either by a tethered towfish equipped with a side-scan sonar, [35], or by using remotely operated vehicles (ROVs) or autonomous underwater vehicles (AUVs), [36]. Deploying a tethered towfish from a boat requires hiring a boat and its crew: a towfish operator, a side-scan operator, and of course experts in the scientific field for which the survey mission is being conducted.

Towfish missions are executed mostly in three phases. The first phase is a general survey mission at higher altitudes, whose goal is to identify possibly interesting parts of the region being covered, [35]. In this phase, the boat tows a towfish in a LM pattern to cover as much area as possible, without or with a low percent of overlapping side-scan sonar swaths. The second phase has the goal of getting detailed low altitude side-scan sonar images of the manually tagged possibly interesting areas from the first phase with side-scan sonar's swaths overlaid to remove any sonic shadows, [35]. The third and final phase includes deploying and Remotely Operated underwater Vehicle (ROV) and/or an AUV, and capturing the identified interesting locations with its high-resolution camera, as proposed in [36]. On the other hand, it would be much more convenient and less costly to deploy an AUV and let it autonomously scan the given area, executing all three above-mentioned area survey phases. It should gather more information about parts of the coverage area that it finds interesting for the current exploration mission, and lower resolution general survey data about other parts of the
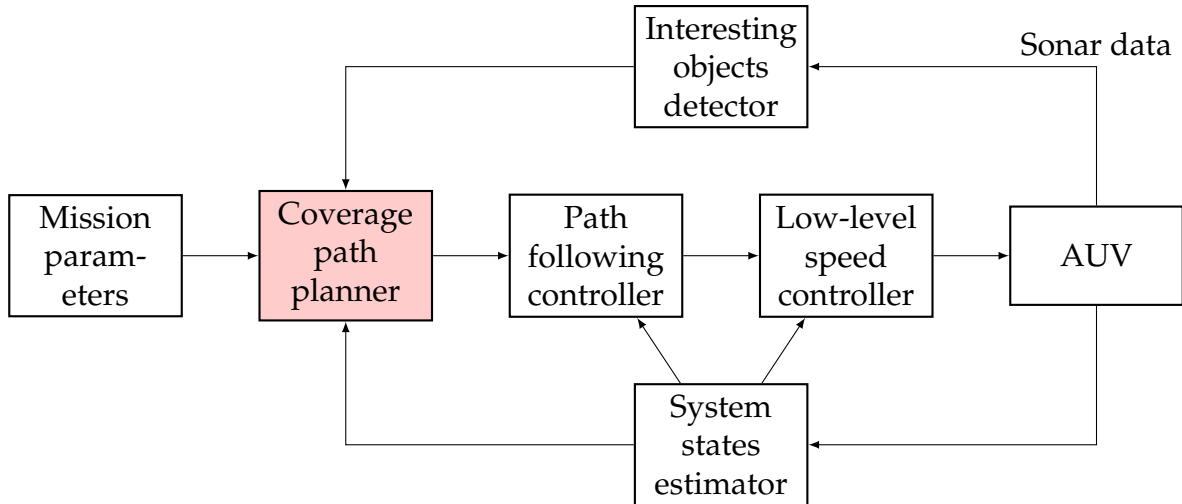
Figure 3.1: Block scheme of the entire AUV system.

area.

Our research goal was to develop an online side-scan sonar data-driven coverage path planning (Coverage Path Planning (CPP)) algorithm for monitoring and surveying large-scale (over $1km^2$) seafloor regions by an AUV, which would replace all three above-mentioned phases of survey missions when using a towfish. Lawnmower pattern is one of the most commonly used solutions for the 2D coverage of an area to be scanned, [37,38]. In marine robotics, it is typically used as a reference path for marine vehicles equipped with side-scan sonar for surveying missions, [35], or as a coverage pattern for vehicles equipped with other sensors, [38].

After designing a mid-level MPC-based surge-yaw controller in Section2.2, the next step towards the above-mentioned research goal was to design an algorithm for solving an online 2D coverage problem, which is a mix between the first and the second mission with a towfish. The idea is to start the mission without overlapping sonar swaths, i.e. with LM lanes twice wider than the sonar range. During the mission, in case that the algorithm detects something interesting in the currently traversed LM lane, coverage path planning algorithm replans the rest of the mission in order to sonify these interesting objects from the opposite side as well, thus gathering more information about the morphology of the interesting objects. Also, it is necessary to develop a control layer for the marine vehicle, in order for it to be able to track the path which the CPP algorithm computes. The block scheme showing the overall view of the system with CPP module denoted as the "Coverage path planner" (in red) is given in Fig.3.1.

Main contributions of the proposed algorithms are: (1) an online side-scan sonar data-driven dynamical programming- and heuristics-based CPP algorithms for large-scale unknown terrains presumably containing relatively few interesting parts. The

algorithm covers the area at best with a twice shorter coverage path than the classical static overlap-all-sonar-swaths LM coverage maneuver (from here on abbreviated as the Classical static overlap-all-sonar-swaths coverage maneuver (CL-CPP) method), (2) analytical upper and lower bounds on performance of the above-mentioned CPP algorithms, validated through extensive mission parameters variation simulations, (3) statistical performance analysis of the proposed CPP algorithms w.r.t. the coverage path length compared to the nonadaptive static solution of overlap-all-sonar-swaths LM coverage maneuver CL-CPP method, and (4) a methodology for approximately choosing survey mission parameters based on coverage area topology and expected probability of coming across interesting objects in the given area. This can be of good use to system engineers operating survey missions with a marine vehicle. Parts of this chapter were previously published by the author in [6,19,39–41].

The rest of this chapter is organized as follows: an overview of sensor data-based path and more specifically coverage path planning algorithms applied for ground, aerial, and especially marine vehicles is given in Section 3.2. Assumptions and definitions which are made for the proposed coverage path planning algorithms in this thesis are given in Section 3.3. A generic formulation of how the coverage path planning algorithms based on side-scan sonar data acquired online should behave is given in Section 3.4. A dynamic programming side-scan sonar data-based algorithm, which is an extension of the existing Wavefront Roughness-based Navigation Function algorithm used for path planning on rough terrains for ground vehicles, is presented in Section 3.5. Section 3.6 introduces a heuristics-based side-scan sonar data adaptive coverage path planning algorithm named Basic Accordion Coverage Path Planning (BA-CPP) algorithm. The following sections, namely Section 3.7 and Section 3.8 propose extensions and improvements of the BA-CPP algorithm. The performance of all four proposed algorithms is analyzed based on extensive mission parameters' variation simulations. These performance metrics are summarized in Section 3.9. Section 3.10 concludes this chapter.

3.2 Coverage path planning algorithms in marine robotics

For a vehicle to reach some predefined goal position, starting from some arbitrary position, while at the same time avoiding the obstacles, it has to use some kind of path planner to successfully execute this task.

There exist several concepts when approaching the path planning problem in marine robotics. Some approaches use optimization techniques to improve the path of the vehicle while avoiding the obstacles, namely: constrained optimization and semi-

infinite constrained optimization, [42], sequential quadratic programming (Sequential quadratic programming (SQP)) with constructive solid geometry (Constructive Solid Geometry (CSG)) representation of the obstacles,[43], or mixed integer linear programming for adaptive sampling of the ocean measurements, [44].

Another option is to use Virtual Force Fields (Virtual Force Fields (VFF)) as in [45]. One of the algorithms which are being used really often in the robotics field is a relatively simple and fast sampling-based Rapidly Exploring Random Tree (Rapidly-exploring random tree (RRT)) algorithm, and its RRT* extension, [46]. Metaheuristic algorithms have also been used to solve the time-energy-optimal path planning problems for marine vehicles, such as genetic and evolutionary algorithms in [47,48], and also ant colony algorithm in [49]. Grid-based path planning algorithms have also been used, i.e. A^* algorithm for minimum energy consumption path in strong currents environments in [50], Fast Marching* (FM and FM*) algorithms for maximum turning rate constrained path generation with a multiresolution method to speed up the path planning process in [51], and Sliding Wavefront Expansion (Sliding Wavefront Expansion (SWE)) Dijkstra-like algorithm for generating minimum duration paths for the AUVs in strong currents, [52]. A comprehensive overview of the path planning algorithms used in marine robotics is given in [53].

Solutions to problems similar to the one defined in Subsection1.2can be found in ground and aerial robotics literature, e.g. in [54–57] and [58]. An offline camera-based LM-like coverage path planning (CPP) algorithm is proposed in [54] as an optimization framework for UAVs monitoring areas of different uncertainty levels which are assumed to be known prior to mission start. A multi-vehicle camera-like sensor-based convergent CPP algorithm was proposed in [55], and further extended with receding horizon (Receding horizon control (RHC)) control level in [59]. A similar concept of exploring different parts of coverage area in different detail, while minimizing coverage path, was the motivation of research in [56], where no more than $0.016km^2$ was covered. The authors had the advantage of using a Unmanned aerial vehicle (UAV) with a camera, which could assess from higher altitude in which way to decompose the area (no more than $0.016km^2$) into cells and optimize the order in which to visit interesting areas and cover them by using LM pattern.

The problem of visual data being occluded from UAV's camera view due to different height of covered objects, which is a problem similar to sonic shadows, is explored in an offline CPP vehicle routing problem-based approach for areas of no more than $0.0625km^2$ in [57]. A so-called modified Zamboni pattern for UAVs, which solves CPP offline and gives a solution with as less sharp turns as possible, is presented in [58], for areas no more than $40 \times 15[m]$,

In marine robotics literature, one of the first papers to mention AUVs used for an online 2D/3D coverage path planning (CPP), which uses camera data in order to generate a photomosaic, is [60]. It guarantees complete coverage and uses standard LM pattern to cover cell-decomposed coverage area. In [61] an online CPP algorithm generates an adaptive but constant-width LM pattern in order to remove the gaps in measurements, for coverage areas no more than $0.48km^2$. The twice wider LM pattern is used only as a limit case solution when no disturbance is present. An online information/entropy based CPP approach for AUVs with side-scan sonar is proposed in [62] with application in target localization. It plans paths which reduce the expected entropy of the surrounding environment w.r.t. coverage path length and total turning angles, on areas of approx. $0.04km^2$. Target (hydrothermal vents) localization online adaptive CPP approach with temperature spreading model and multi-vehicle cooperation on areas of $1.4km^2$ is presented in [63].

An online approach to mapping of ever-changing marine habitat by an AUV equipped with a camera and multi-beam sonar is given in [64]. It covers arbitrarily identified regions of interest (ROIs), area of $0.0375km^2$ from previous mapping missions in such a way to minimize repeated coverage. The accent of this approach is more on Region of interest (ROI) identification and coverage path minimization through intra-ROI path length minimization, not adapting the standard LM pattern inside each decomposed cell. Another sonar-based survey path planning approach in [65] uses a-priori known bathymetric data of the coverage area of approx. $0.06km^2$ in order to find salient points and cover them while minimizing vehicle position and sensing uncertainties. In [66] side-scan sonar data is improved for search missions through adapting the width of LM lanes w.r.t. pose estimation and sonar data uncertainty. A comprehensive overview of CPP methods, with an emphasis on their application in underwater robotics, is given in [38]. A solution to the problem of LM lane visiting order, in the case when lanes are narrower than the minimum turning radius of the vehicle, is presented in [67].

Majority of the above-mentioned CPP methods use a camera-like sensor, while side-scan sonar sensing suffers from the problem of sonic shadows behind some objects. Also, in case they reconfigure LM pattern, they do it to minimize the number of turns or the total turning angle. Otherwise, they generate paths which can be shorter than LM pattern path but are not well suited for side-scan sonar applications. Moreover, only one of the above-mentioned CPP algorithms tackles the problem of large-scale areas of over $1km^2$, and the rest do not give performance guarantees in case of coverage area scaling up.

3.3 Assumptions and definitions

Definitions of informational gain metrics

Let $\mathcal{T}$ denote a sea floor terrain (backscatter intensity) map which contains m rows and n columns of square patches p_{ij}, whose size is dependent on the resolution of the used sonar, and assume that the map $\mathcal{T}$ is not a-priori known. Information gain in the context of exploring the sea floor can be defined as the roughness of the sea floor, i.e. the local variance of the side-scan sonar data.

Much of the roughness metrics research is related primarily to the Autonomous Ground Vehuicle (AGV)s. In [68] the roughness metrics is based not only on the local variance of the terrain map in the spatial domain, but it also takes into account terrain's intrinsic properties in the frequency domain. Several spatial measures of the terrain's roughness are presented in [69]. In marine robotics, roughness is mostly related to the exploration of the seafloor relief complexity, [70], and also in the research related to identifying benthic habitat types, [71].

Roughness value $r_{i,j}$ represents the informational gain measure of a map patch $p_{i,j}$ (how interesting or informational that patch is) and it is calculated as a standard deviation of the terrain map values of its neighboring map patches, [72]. $\mathcal{N}_{i,j}$ denotes a set of 8-connected neighborhood patches of the patch $p_{i,j}$ including the patch $p_{i,j}$ itself. Roughness value $r_{i,j}$ is now defined as:

$$r_{i,j} = \mu_{i,j}\sqrt{\mathrm{Var}(\mathcal{T}(\mathcal{N}_{i,j}))}, \tag{3.1}$$

where $\mu_{i,j} \in [0,1]$ is a coefficient dependent on the terrain type of the map patch $p_{i,j}$, and Var denotes the variance operator.

In order to remove the slope of the terrain out of the roughness measure, i.e. to remove the linear trend out of the sampled elevation values, it is possible to redefine roughness as the standard deviation of the residuum values $\rho_{i,j}$ w.r.t. the interpolated plane through the patches $\mathcal{N}_{i,j}$:

$$r_{i,j} = \mu_{i,j}\sqrt{\mathrm{Var}(\rho(\mathcal{N}_{i,j}))}. \tag{3.2}$$

Considering that the main idea of the research is to navigate the vehicle along the way of maximum informational gain, i.e. roughness or minimum smoothness, it is necessary to make a conversion of the roughness into smoothness. This way, after propagation the smoothness costs in the dynamical programming-based coverage path planning algorithm, a minimum-smoothness or maximum-information-gain path is

obtained. Smoothness can be defined as:

$$s_{i,j} = \begin{cases} 1, & \text{if } r_{i,j} = 0 \\ 1/r_{i,j}, & \text{otherwise}. \end{cases} \tag{3.3}$$

Apart from defining the informational gain of side-scan sonar images as the local variance of the backscatter data, which is heavily sensitive to noise, informational gain can be defined as the output of some saliency/anomaly detection method applied to side-scan sonar imagery. Overview of such methods, and the methodology of choosing the most appropriate one for use with side-scan sonar data are given in Chapter 4.

Assumptions related to lawnmower pattern

Let us denote LM line length by Lm, width between LM transects by Wm, and a parameter which denotes the ratio between LM segment length and width, denoted by α_{lm}:

$$\alpha_{lm} = \frac{W}{L}. \tag{3.4}$$

Furthermore, parameter W depends on side-scan sonar's horizontal range $w_{sss}m$ which an AUV operator sets, i.e. $W = f(w_{sss})$. Generally, LM segments are set in such a way that $\alpha_{lm} \ll 1$ holds, i.e. LM segments are much more elongated than its segments are wide. Extreme cases when $\alpha_{lm} > 1$ are also taken into account to test the proposed CPP algorithm's performance.

Assumptions related to AUV

Desired surge speed is assumed to be constant during the whole mission in order to get equidistant along-track sonar beams, and is denoted as $u[m/s]$. Desired constant altitude at which AUV is commanded to operate is denoted by $h_{ref}[m]$. Reference yaw rate $r_{ref}[rad/s]$ is computed by a high-level model predictive controller (MPC) and is tracked by a low-level PID-type controller.

Control side of mission execution is out of the scope of this thesis. Altitude control is decoupled from $x - y$ position control. It is assumed that external disturbance to the vehicle is either not present, or properly rejected by the model predictive (MPC) path following controller presented in [7]. The proposed CPP algorithm assumes the vehicle to be a point mass without any dynamic behavior, so control level has the task to follow paths which the CPP algorithm generates. It is also assumed that localization is either perfect or improved by communication of the AUV with an autonomous surface vehicle (Autonomous Surface Vehicle (ASV)) as in [73], or a fixed beacon(s) as in [74]. Hence, the proposed approach does not minimize the number of turns during

the mission, as does [75], in order to reduce dead reckoning error which increases the most during turning maneuvers.

The blind spot of the side-scan sonar (*nadir*), is assumed not to be present (which holds if nadir is covered with an additional multibeam sonar).

Assumptions related to sea floor configuration

The area which the vehicle should cover is assumed to be relatively sparsely populated with objects of interest for a given mission. Definition of "interesting" could vary from one mission to another, that is from one research area to another.

It could be a measure of seabed rugosity/roughness in coral and seabed relief exploration or marine archeology, a measure of the spectral density of return signal to echosounder in Posidonia seagrass habitat research, the output of some object recognition and/or classification algorithm in marine archeology, security threats, or marine biology. Here, information gain is defined as a measure of local side-scan sonar data variation.

Moreover, it is assumed that the sea floor is approximately flat. In case that the seafloor terrain is slanted dominantly along one direction, then the lawnmower pattern mission should be oriented parallel to the terrain's principal gradient. Since large-scale open sea areas to survey with side-scan sonar are of authors' research interest, obstacles are not assumed to be present in the environment.

3.4 Generic formulation of the proposed coverage algorithms

The initial solution of the proposed CPP algorithm is an LM pattern with lanes twice as wide as sonar range. This is, of course, a good solution only if there is nothing of interest for the mission in the area covered.

In case that the vehicle detects some interesting object(s) in its sonar data while traversing the current LM line leg, the coverage path planning algorithm should replan the rest of the mission in such a way to sonify the interesting object(s) from the opposite side. Also, it should again (optimistically) assume that during the rest of the mission it will not encounter any interesting objects, and thus it will again generate "stretched" LM pattern for the remaining part of the mission. Moreover, in the limit case when the sea floor area of interest is highly covered interesting objects, then the proposed coverage path planning algorithm should behave as the CL-CPP.
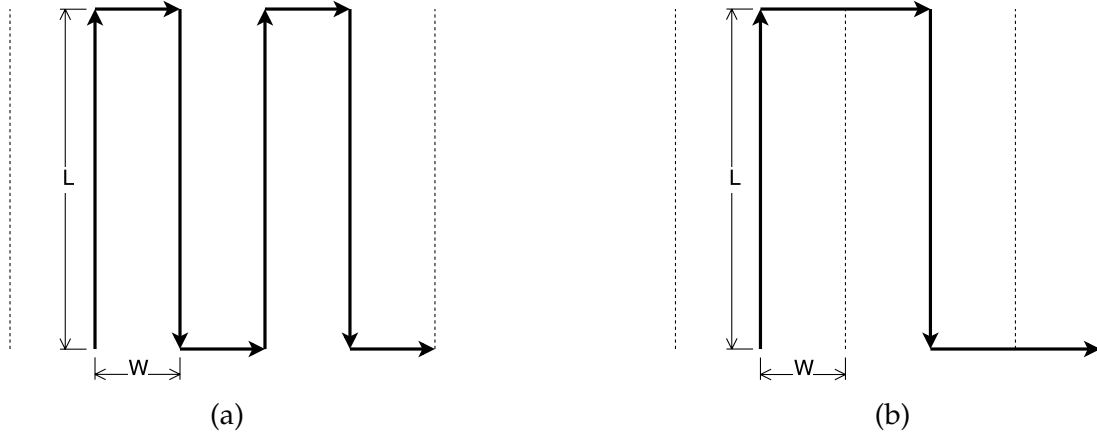
Figure 3.2: Elementary period of the LM pattern for side-scan sonar survey missions. (a) CL-CPP pattern. (b) The initial solution of the proposed CPP algorithm.

Since the proposed CPP solution is mostly a regular pattern, at least in the initial moment of the mission, elementary periods of the CL-CPP and the proposed CPP approach are given in Fig.3.2aand3.2b, respectively. This is needed to analytically predict the behavior of the CPP algorithms proposed in this thesis on a spatial period of width $4W$, which ends with a horizontal line. This way every LM pattern spatial period starts with a vertical line and is of the same width.

Heuristics behind the area covering of CL-CPP, as well as finite state machine of the proposed CPP algorithm, ensure that complete coverage is achieved by the end of the mission, under the assumptions stated in Section3.2. That is why it is not required that the algorithm achieves a full coverage of the current pattern spatial period, but instead to achieve globally complete coverage in terms of acquiring all informative data present within the whole coverage area.

3.5 Wavefront Roughness-based Navigation Function Coverage Path Planning Algorithm

This section gives a short overview of the authors' prior work on a path planning algorithm named WRbNF algorithm, [19], and describes how it has been extended for use in the CPP missions, thus proposing the Wavefront Roughness based Navigational Function Coverage Path Planning (WRBNF-CPP) algorithm. Since Wavefront Roughness-based Navigation Function plans a path from vehicle's current position to some goal position based on cost propagation, a proper cost function needs to be defined in order for the algorithm to behave the way it is supposed to. The original WRbNF algorithm was designed for path planning on rough terrains, i.e. to navigate the vehicle along the least rough terrain. In the context of side-scan sonar data-based

seafloor exploration, the algorithm actually needs to guide the vehicle towards places which are interesting, i.e. which have greater informational gain for the exploration mission. In that way, the algorithm stayed the same, but the concept of cost function was inverted.

Various terms in the cost function and their respective weights were tested to see which combination fit the side-scan sonar exploration mission the best. Also, analytical analysis of coverage performance w.r.t. its coverage path length was conducted and verified by extensive simulations with mission parameters variation. This algorithm was proven to perform better than the CL-CPP algorithm for most of the mission parameters' values which are assumed in this dissertation. The results of WRBNF-CPP algorithm are published in [40].

3.5.1 Wavefront Roughness-based Navigation Function

Wavefront Roughness-based Navigation Function (WRbNF) algorithm, [19], is an approximation of an optimal cost-to-go map, which estimates the traversability measure from every location on the rough terrain to the goal position, and is an extension of the Roughness-based Navigation Function (RbNF) algorithm, [76]. The (W)RbNF computation is much easier than in the case of Dijkstra algorithm. The results show that the majority of the computed values of the WRbNF map are near optimal regardless of the terrain roughness, number and size of the obstacles. Since the WRbNF algorithm has an inherently parallel structure, it can be coded to significantly outperform the Dijkstra algorithm in terms of runtime by using an adequate amount of computer resources, [19]. An example of the WRbNF algorithm used for generating a cost-to-go map on a large scale rough terrain with obstacles is given in Fig.3.3.

The reason for using such an algorithm in coverage path planning missions was to allow the path planner to autonomously lead the vehicle towards the most information-rich areas of the sea floor, the same way WRbNF cost-to-go map leads the vehicle along the least rough terrain.

3.5.2 Design of cost function

One of the first tasks that had to be solved in order to use WRbNF algorithm for the CPP problem defined in Subsection1.2was to define which part of the obtained side-scan sonar data will be used as the cost map for the current LM lane about to be covered, e.g. lane $x \in [100, 150][m]$, and $y \in [0, 500][m]$ shown in Fig.3.4a. The solution is to use the (simulated) sonar data from the previous sonar swath's half closer to the current LM lane, i.e. area $x \in [50, 100][m]$, and $y \in [0, 500][m]$ in Fig.3.4a. This means
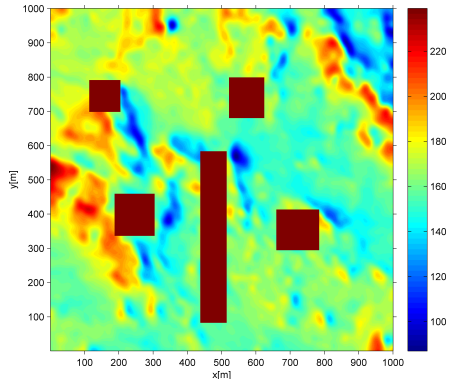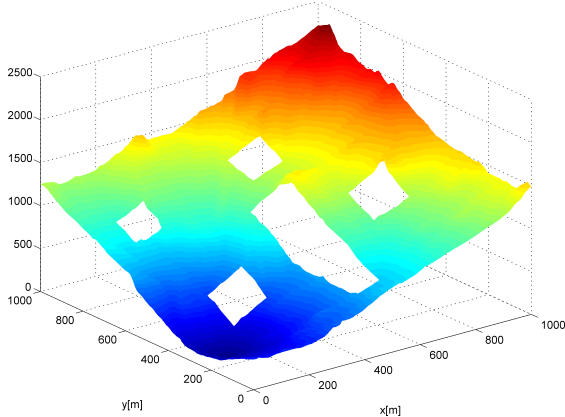
(a) An example of the terrain map.
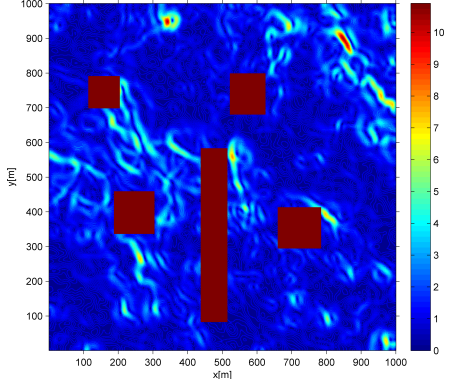
(b) Roughness map computed based on the terrain map.
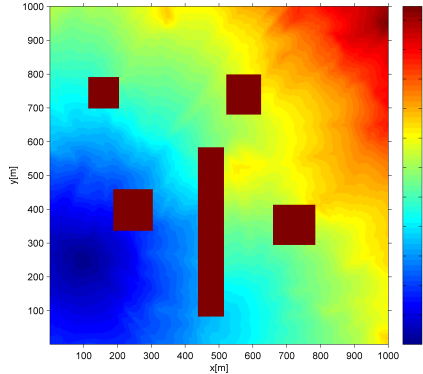
(c) 2D projection of the WRbNF cost-to-go map. Costs are propagated from the goal position $(100, 250)$.

(d) 2.5D manifold representing the cost-to-go WRbNF map. Costs are propagated from the goal position $(100, 250)$.

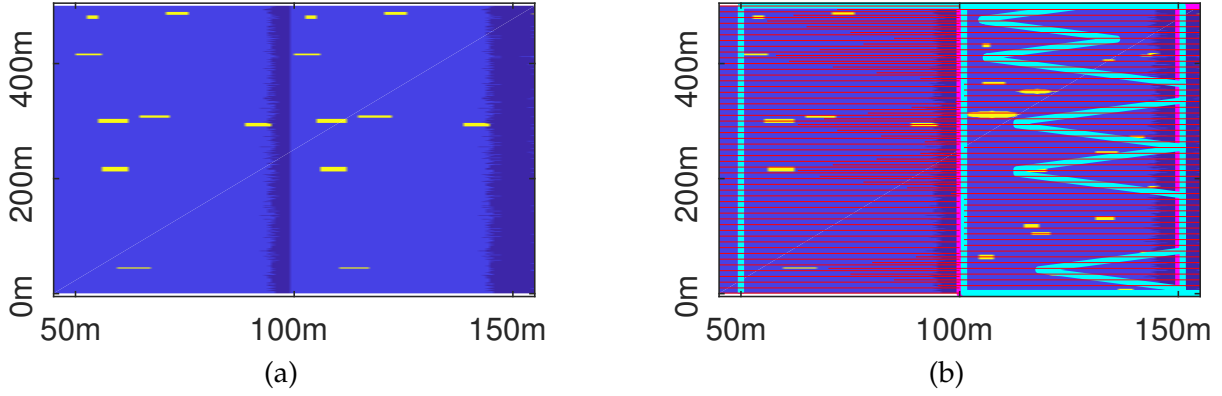**Figure 3.3:** An example of the WRbNF algorithm's performance.

Figure 3.4: (a) An example of the side-scan sonar cost map replication for the current LM lane. Vehicle's path in the LM lane $x \in [100, 150][m]$, and $y \in [0, 500][m]$ is planned based on the sonar data from the previous sonar swath's $x \in [50, 100][m]$, and $y \in [0, 500][m]$. (b) An example of the coverage path planning based on previous swath's sonar data. (cyan) path of the vehicle following the informational gain-guided CPP solution, (dark blue) low informational gain areas, (yellow) high information gain objects, (red) side-scan sonar range.

that the vehicle locally plans a path to go along the most informative path based on the cost map of the previous swath's half, but globally it moves in the current LM lane, at a distance of half a swath away from the interesting objects, thus removing their sonic shadow. The desired behavior of the CPP algorithm is depicted in Fig.3.4b, where the vehicle moves along the planned path (cyan line) in such a way to sonify (red lines depicting side-scan sonar ranges) interesting objects' sonic shadows from the opposite side as well (side-scan sonar simulated range lines reaching all the way to the right side of the interesting objects).

The second task addressed in order to use WRbNF algorithm described in Subsection3.5.1was to design such a cost function, which would reflect the desired exploratory behavior of the coverage path planner. It is supposed to plan a path which maximizes along-path information gain for the mission at hand. In Subsection3.3it was mentioned that informational gain in the context of maritime exploration by a side-scan sonar could be modeled as the smoothness of the sea floor terrain. The cost function of the WRbNF-based coverage path planning algorithm is thus defined as:

$$c_{i,j} = s_{i,j} + \alpha_d d_{line_{i,j}} + \beta_{WP} d_{WP_{i,j}} \tag{3.5}$$

where $s_{i,j}$ is the smoothness defined by Eq. (3.3), $d_{line_{i,j}}$ is defined as a distance of map patch $p_{i,j}$ in the cost map from the current lawnmower line being followed (the left or right edge of the map, depending on whether the vehicle moves upwards or downwards in the local cost map), and $d_{WP_{i,j}}$ is the distance of each map patch $p_{i,j}$ from the currently set waypoint WP in the local cost map frame.

3.5.3 WRbNF propagated from a lawnmower goal waypoint

WRbNF propagates costs from the current LM goal Waypoint (WP) outward in a radial fashion, through square-shaped rims, as described in [19], see Fig.3.3d. This goal WP is located either in the top-right, bottom-right corner of the cost map defined in Subsection3.5.2, depending on whether the vehicle's current LM lane stretches upwards or downwards in the cost map.

WRbNF with a goal WP of cost propagation chosen as the current LM WP (see Fig.3.5a) has been proven to be of little use for CPP problem defined in Subsection1.2. The main drawback of this approach is the way the goal WP is chosen. Neither extensive tuning of parameters α_d, nor β_{WP}, nor extremely low cost values of arbitrarily placed interesting objects in the cost map, nor negative costs of interesting objects (due to local minima and saddle points in the cost-to-go map) helped the WRBNF-CPP algorithm to significantly diverge from the line if something interesting has been spotted in the previous LM lane.

The cause of this problem has been discovered to be the way in which WRbNF is propagated radially through square-shaped rims, but not along rims as well. This leads to low cost (interesting) objects placed in inconvenient places in the cost map, not "pulling" the WRbNF planner to diverge from the current LM line towards them. A solution to this problem is presented in Subsection3.5.4.

3.5.4 WRbNF propagated from a dislocated goal waypoint

The novelty that is introduced in this approach is that the propagation of costs in WRbNF is not done from the currently followed LM WP, but from a point outside the cost map. This was done in order to eliminate the problem described in Subsection3.5.3. Coordinates of this point in the current LM lane are given by:

$$x_{WP_{new}} = W/2, \tag{3.6}$$

$$y_{WP_{new}} = \begin{cases} -W/2, if\,downward\,motion \\ L + W/2, if\,upward\,motion. \end{cases} \tag{3.7}$$

Graphical representation of such WP choice is given in Fig.3.5b. This way the costs are propagated row-by-row upward or downward through the cost map, depending on the movement direction of the vehicle. The starting point of the vehicle in the generated cost-to-go map is not the LM WP at the LM lane's beginning, but a minimum cost-to-go point on the topmost/bottom most row of the cost-to-go map, depending on the movement direction of the vehicle. Analogously, the end point of vehicle's path
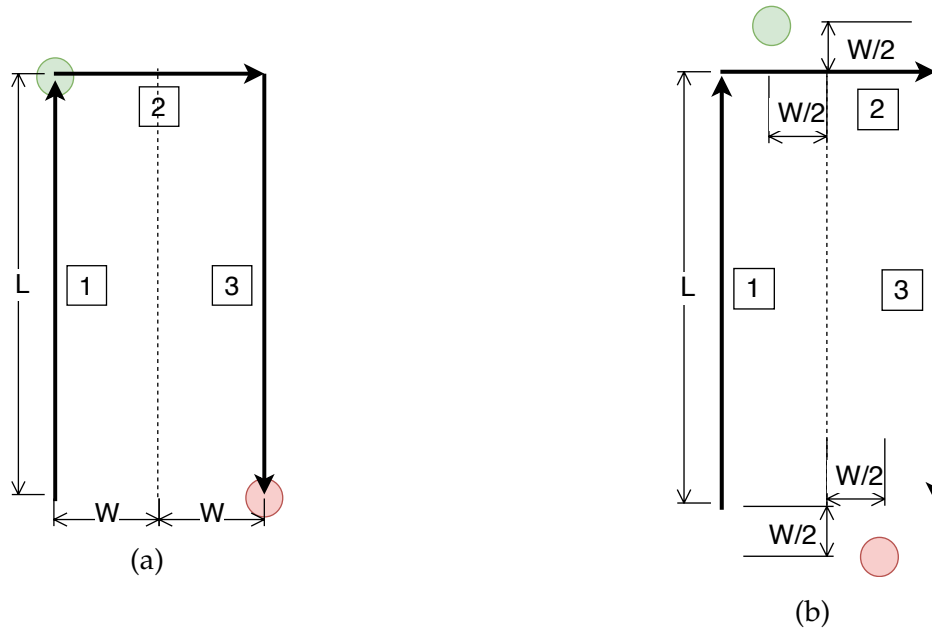
Figure 3.5: Choices of the goal position for cost propagation by the WRbNF algorithm for CPP purposes. (a) Lawnmower pattern corner WP. (b) Goal WP dislocated out of the LM lane. (green) waypoint generated for "upward" motion, (red) waypoint generated for "downward" motion.

in the cost map is a minimum cost-to-go point in the bottom most/topmost row of the cost map. The starting point has to be computed, and the ending point is reached by simply moving down the WRbNF cost-to-go map.

Cost function parameters were set to the same values as when the WRbNF was propagated from the current LM WP, i.e. $\alpha_d = 0$, and $\beta_{WP} = 0$.

3.5.5 Behavior of the proposed algorithm

The initial solution of the WRBNF-CPP algorithm is an LM pattern with lanes twice as wide as sonar range, see Fig.3.6a. This is a good solution only if coverage area is devoid of any interesting objects. It is important to note that the below presented behavior of the WRBNF-CPP algorithm is analogous for cases when the vehicle movies in the opposite direction (upward/downward) of the directions in the cases described below.

If the vehicle detects something interesting to its right in the cost map, as it does while moving along line 1 in Fig.3.6b, it will plan the path 3 (using WRbNF algorithm on cost map based on right half of side-scan sonar's swath along LM line 1) in order to sonify interesting objects' sonic shadow(s) from the opposite side.

If the vehicle detects something interesting to its right in the cost map, as it does while moving along line 3 in Fig.3.6c, it will generate new WPs in order to perform a loop back to its right side (lines 4, 5, 6, and 7), and thus sonify interesting objects' sonic

shadow(s) from the opposite side. In order to start the next LM pattern spatial period, this CPP method will generate another LM line 8.

If the vehicle detects something interesting to its right in the cost map as while moving along line 1 in Fig.3.6d, but it also detects something to the left/right while moving along the planned WRbNF path 3, it will again generate new WPs in order to perform a loop back to its right side (lines 4, 5, 6, and 7), and thus sonify interesting objects' sonic shadow(s) (that it made in the current LM lane 3) from the opposite side. In order to start the next LM pattern spatial period, LM line 8 is generated.

If the vehicle detects something interesting to its left in the sonar image, as it does while moving along line 3 in Fig.3.6e, it will plan the path 5 (using WRbNF algorithm on cost map based on left half of side-scan sonar's swath along LM line 3) in order to sonify interesting objects' sonic shadow(s) from the opposite side.

In case that the vehicle detects interesting object(s) to its right in the cost map, as it does while moving along line 1 in Fig.3.6f, and then detects interesting object(s) to both its left and right side while moving along path 3, the proposed WRBNF-CPP algorithm will again generate new WPs in order to perform a loop back to its right side (lines 4, 5, 6, and 7), then line 8, and plan path 9 based on the cost map from the previous sonar range.

Table3.1shows path lengths for all the above-mentioned cases in one LM pattern spatial period, which are based on graphs depicted in Fig.3.6a-3.6f. The length of the longest WRbNF generated path which diverges from the current LM line, as e.g. in Fig.3.6b,3.6d,3.6e, and3.6f, is derived from the way in which WRbNF propagates its costs. In case that there are interesting objects in the previous swath, placed in such a way that WRbNF path forms consecutive isosceles right triangles, then the maximum length of such WRbNF path L_{WRbNF} is given by $L_{WRbNF} = \sqrt{2}L$.

3.5.6 Analysis of the upper and lower performance bounds

In this subsection the best- and the worst- scenarios for the WRBNF-CPP performance are analyzed in order to predict upper and lower bounds of its performance. A performance measure $e_{CL-CPP}^{WRbNF-CPP} e_{CL-CPP}^{WRbNF-CPP}$ is chosen to represent relative improvement of the WRBNF-CPP generated coverage path $l_{WRbNF-CPP}$ w.r.t. the CL-CPP path length l_{CL-CPP}. In case that the sea floor area which is being explored is devoid of any interesting objects (see Fig.3.6a), then the *best-case scenario* improvement which the WRBNF-CPP has over the CL-CPP algorithm, i.e. the upper bound on $e_{CL-CPP}^{WRbNF-CPP}$, is given by:
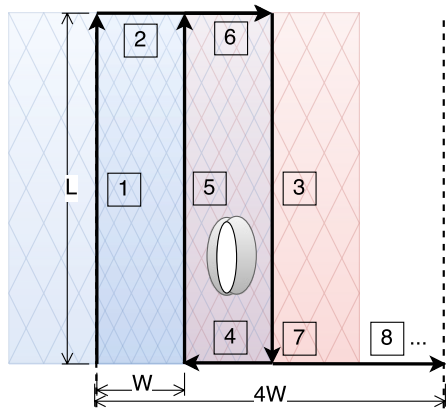
$$e_{CL-CPP}^{best\,WRbNF-CPP} = \frac{1}{2(1+\alpha_{lm})}100[\%] \tag{3.8}$$
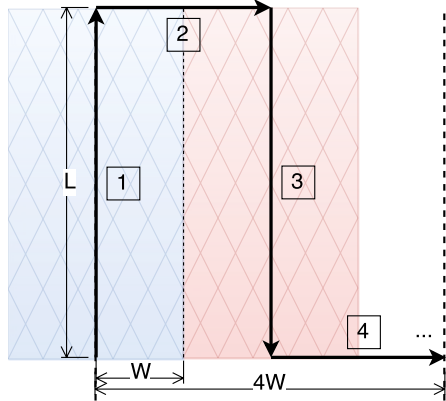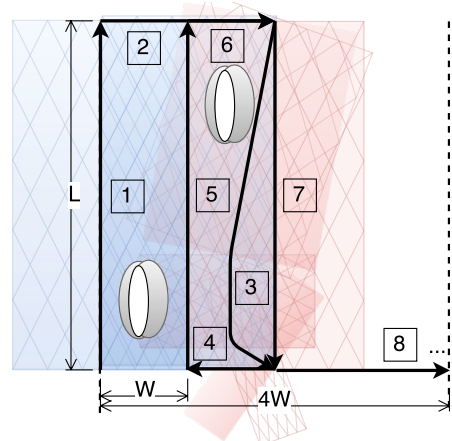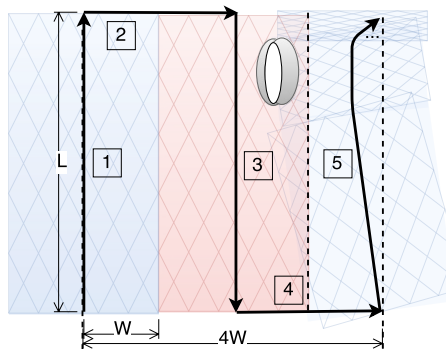
(a) Case 0: sea floor is completely barren
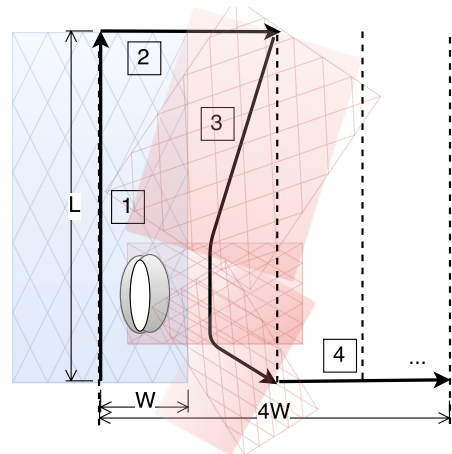
(b) Case 1: something was detected in the previous swath

(c) Case 2: nothing is detected in the previous swath and an interesting object was detected in the current swath to the left of the current lawnmower leg

(d) Case 3: something was detected in the previous swath, but also to the left of the vehicle's current path

(e) Case 4: something is detected to the right of the vehicle path in the current swath
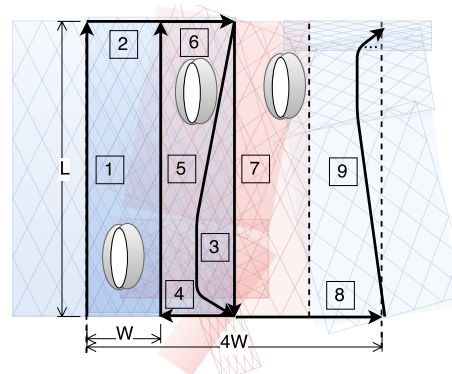
(f) Case 5: something is spotted both to the left and right of the vehicle's current path

Figure 3.6: Characteristic cases for specific placement of interesting objects w.r.t. the previous left, or right sonar swath.

Table 3.1: Path lengths for different specific cases of the WRBNF-CPP algorithm behavior

Case No.	Path length of the CL-CPP method	Path length of the WRBNF-CPP method
0	$4L + 4W$	$2L + 4W$
1	$4L + 4W$	$\leq L(1 + \sqrt{2}) + 4W$
2	$4L + 4W$	$5L + 6W$
3	$4L + 4W$	$\leq L(4 + \sqrt{2}) + 6W$
4	$4L + 4W$	$\leq L(2 + \sqrt{2}) + 4W$
5	$4L + 4W$	$\leq L(3 + 2\sqrt{2}) + 6W$
5'	$4L + 4W$	$\leq L(4 + 2\sqrt{2}) + 8W$

If parameter $\alpha_{lm} \to 0$, the improvement of the WRbNF-CPP method tends to $e_{CL-CPP}^{best\,WRbNF-CPP} \to E_{best}^{WRbNF} = 50\%$, ensuring a twice shorter coverage path than in case of CL-CPP. On the other hand, if $\alpha_{lm} \to \infty$, the relative improvement tends to go to $e_{CL-CPP}^{best\,WRbNF-CPP} \to 0\%$.

In case interesting objects are positioned in such a way that the vehicle has do to a maneuver shown in Fig.3.6f, with line 1 not being straight, but a diverged path planned by WRbNF algorithm based on previous LM lane's sonar data (as is line 5 in Fig.3.6c) in each LM pattern spatial period, and then do a loop back as in case of Fig.3.6c, then the worst-case scenario improvement (case 5' in Table3.1) which the WRBNF-CPP has over the CL-CPP method, i.e. the lower bound on $e_{CL-CPP}^{WRbNF-CPP}$, is given by:

$$e_{CL-CPP}^{worst\,WRbNF-CPP} = \frac{-(2\sqrt{2} + 4\alpha_{lm})}{4(1 + \alpha_{lm})}100[\%] \tag{3.9}$$

This means that for $\alpha_{lm} \to 0$, the worst relative improvement tends to go to $e_{CL-CPP}^{worst\,WRbNF-CPP} \to -70.7\%$. On the other hand, if $\alpha_{lm} \to \infty$, the worst relative improvement tends to go to $e_{CL-CPP}^{worst\,WRbNF-CPP} \to -100\%$. It is possible for the values of $e_{CL-CPP}^{WRbNF-CPP}$ to go even below the predicted lower bound in case of multiple loopbacks throughout the whole coverage area.

3.5.7 Simulation results

In order to visually present the WRBNF-CPP algorithm's behavior in simulated environment, a few performance examples are given in Fig.3.7with values of $\alpha_{lm} = 0.01$, and the percentage of sonar ranges in the coverage area containing interesting objects $p_{obj} \in \{0, 20, 40, 60, 80, 100\}[\%]$. It is clear that the WRBNF-CPP algorithm produces, in

terms of mean values, coverage paths which are at best approximately 50% shorter than CL-CPP pattern (see Fig.3.7a). In Fig.3.7c,3.7e, and3.7ffor $p_{obj} \in \{20, 40, 60, 80, 100\}\%$, improvements w.r.t. the CL-CPP method amount to $e_{CL-CPP}^{WRbNF-CPP} \in \{10, -10, -15, -20, -40\}\%$, respectively. As for the graphical representation of the WRBNF-CPP performance, the better WRbNF-CPP performance, the more CL-CPP (magenta) lines are left uncovered, see Fig.3.7cand Fig.3.7e. Worsened WRBNF-CPP performance are equivalent to more CL-CPP (magenta) lines being overlaid by WRBNF-CPP (cyan) lines, due to loop back maneuvers, here notable in Fig.3.7e and3.7f.

3.5.8 Statistical analysis of algorithm performance

In order to gain further insight into the performance of the proposed WRBNF-CPP algorithm, extensive parameter variation simulations have been conducted. Parameter α_{lm} took values $\alpha_{lm} \in \{0.01, 0.1, 0.25, 0.5, 0.75, 1, 2\}$, while p_{obj} had values from 0 to 100% with a 5% step. The width of the sonar range was fixed to $W = 50m$, and thus LM line length was computed as $L = W/\alpha_{lm}$. It is important to emphasize that the performance measure has been chosen as relative w.r.t. the CL-CPP algorithm, thus making the results independent on the absolute values of parameters L and W.

For each possible tuple (α_{lm}, p_{obj}), 100 tests with appropriate information gain cost maps have been generated, where the interesting objects have been dispersed randomly. Coverage area C had values $C \in \{5.5, 3.8, 3.3, 2.73, 2.34, 2.05, 1.36\}[km^2]$ for each α_{lm}, respectively. The value of the coverage area depended on LM parameters L and W, as well as maximum mission duration set to $15h$ at a constant mean surge speed of $1.5m/s$. Cost maps values were binarised in a way that the value of $c = 0.1$ meant noninteresting areas, while $c = 1$ meant interesting areas. Relative coverage path length improvement of the WRBNF-CPP algorithm over the CL-CPP pattern was averaged over 100 tests per each (α_{lm}, p_{obj}) parameter tuple.

The results of this analysis are shown in Fig.3.8a. It can be noted that the average improvement of the WRBNF-CPP w.r.t. the CL-CPP is most significant in the area defined by low α_{lm}, $\alpha_{lm} \leq 0.25$, with $p_{obj} \leq 15\%$. In these cases, the WRBNF-CPP generates coverage paths which are on average $20 - 50\%$ shorter than the CL-CPP paths.

As α_{lm} grows in value, the performance of WRbNF-based CPP algorithm decreases when compared to its estimated performance, e.g. for $\alpha_{lm} = 1$ WRbNF has mean performance of approx. -100%, which is 110% worse than the estimated -47.75%. This is due to the property of WRbNF-based coverage path planning algorithm to plan the path along the current lawnmower segment w.r.t. the cost map generated from
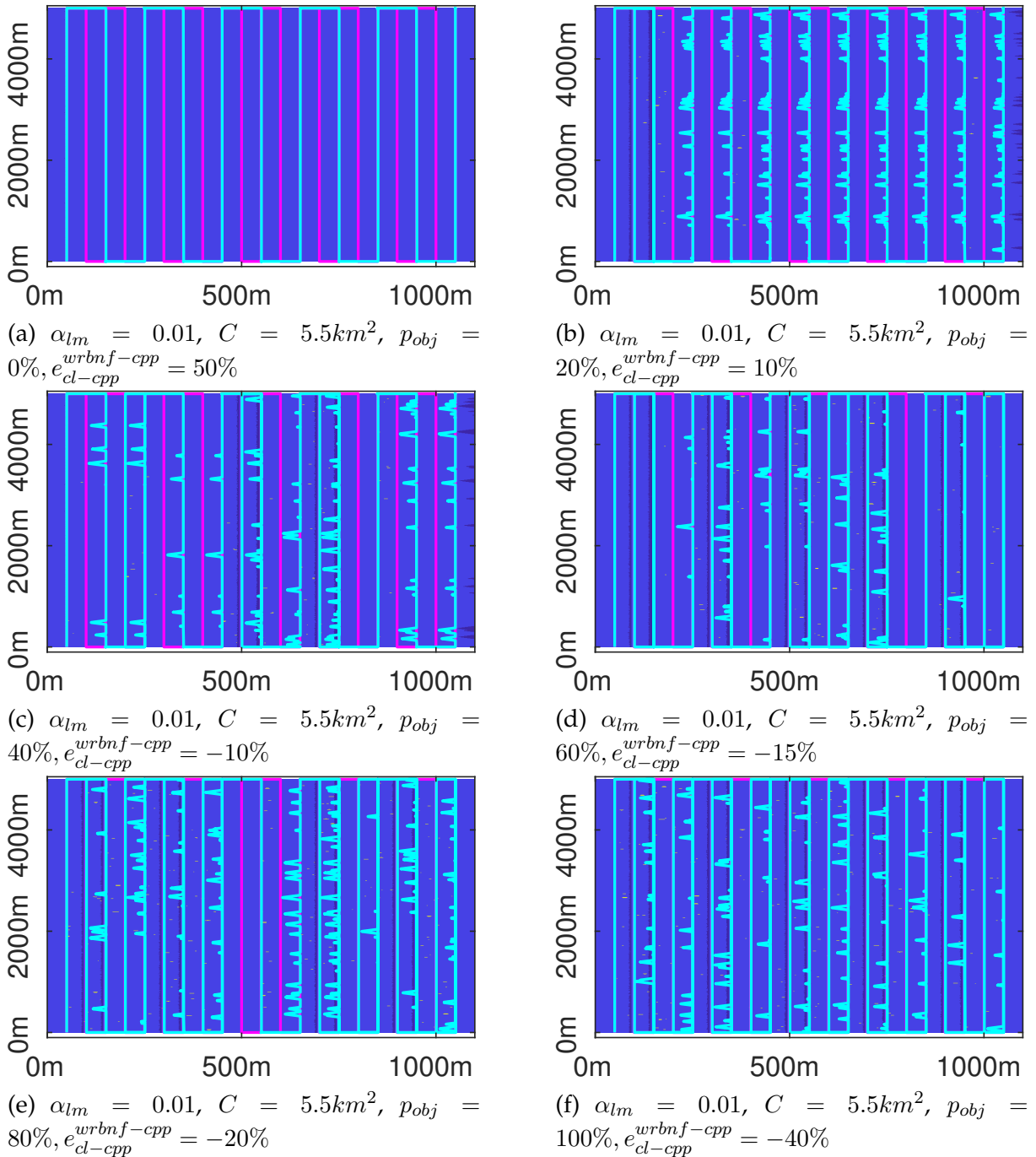
(a) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 0\%$, $e_{cl-cpp}^{wrbnf-cpp} = 50\%$

(b) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 20\%$, $e_{cl-cpp}^{wrbnf-cpp} = 10\%$

(c) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 40\%$, $e_{cl-cpp}^{wrbnf-cpp} = -10\%$

(d) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 60\%$, $e_{cl-cpp}^{wrbnf-cpp} = -15\%$

(e) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 80\%$, $e_{cl-cpp}^{wrbnf-cpp} = -20\%$

(f) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 100\%$, $e_{cl-cpp}^{wrbnf-cpp} = -40\%$

Figure 3.7: Examples of the WRBNF-CPP algorithm performance compared to the CL-CPP pattern, with the visual representation of coverage area and coverage paths. (cyan) path of the vehicle following the WRBNF-CPP solution, (magenta) path of the vehicle following the CL-CPP pattern, (dark blue) low informational gain areas, (yellow) high information gain objects.
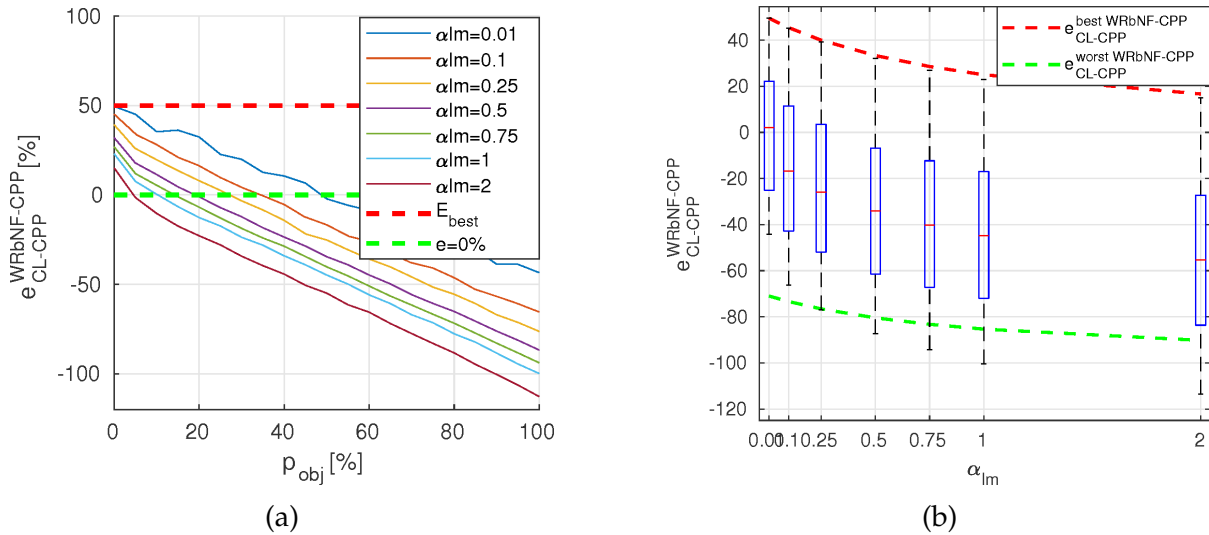
Figure 3.8: (a) WRBNF-CPP algorithm averaged improvements relative to the CL-CPP algorithm. (b) Coverage path length improvements distributions of the WRBNF-CPP relative to the CL-CPP algorithm.

the previous side-scan sonar swath. In case that there is something interesting in one swath, and nothing interesting further along the lawnmower pattern until the end of the mission, the vehicle will diverge to cover sonic shadows of that previous interesting swath, but will in turn diverge from all succeeding lawnmower lines, because it will try to fill in the gap in the previous swath, left initially by the first divergence from the set lawnmower line.

These results are important from a system engineer's perspective since it allows an estimate of possible survey mission duration reduction based on an a-priori known value of α_{lm}, and a rough estimate of p_{obj}.

Best- and worst-case scenario analysis validation

Distributions of relative coverage path improvements of the WRBNF-CPP algorithm w.r.t. the CL-CPP for each fixed α_{lm}, and varied p_{obj}, are shown in Fig.3.8b. Each of the boxplots contains the respective improvement values obtained from 2100 simulations ran per each α_{lm} value. It can be noted that the simulation results of the best-case scenarios match the predicted algorithm performance given by Eq.3.8 almost perfectly. On the other hand, the worst-case performance of the WRBNF-CPP for each α_{lm}, described in Subsection 3.5.6 go even below the predictions given by Eq.3.9, which has been hypothesized in Subsection 3.8b.

Better mean performance probabilities

Fig.3.9ashows the percentage of simulations p_{better}, for each α_{lm}, in which the WRBNF-CPP algorithm generated shorter coverage paths than the CL-CPP algorithm. WRbNF has been shown to have better performance than the CL-CPP in over 35% of the simulations for $\alpha_{lm} \leq 0.1$.

Worst-case performance analysis

In order to gain further insight into the less good improvements of the WRBNF-CPP algorithm w.r.t. the CL-CPP (shown in Fig.3.8a), Fig.3.9bshows the worst-case improvements of the coverage paths generated by the WRBNF-CPP algorithm w.r.t. the CL-CPP method, for parameter α_{lm} taking values $\alpha_{lm} \in \{0.01, 0.1, 0.25, 0.5, 0.75, 1, 2\}$.

It is notable that WRBNF-CPP worst-case relative performance is negative for each value of α_{lm}, which is probably caused by the loop-back maneuvers and further divergences of the coverage path from the initially set LM lines.

Limit case performance analysis

Curves $e_{CL-CPP}^{WRbNF-CPP}(\alpha_{lm}, p_{obj}) \triangleq 0 = p_{obj}^{critical}(\alpha_{lm})$ are shown in Fig.3.9c. This curve represents the limit case in which the performance of the WRBNF-CPP algorithm cannot guarantee to have on average better performance than the CL-CPP method. It can also be generated based on the graph in Fig.3.8aas a curve connecting zeros of all the curves as a function of belonging α_{lm} values.

WRBNF-CPP algorithm has on average equal performance as the CL-CPP algorithm when $p_{obj} \geq 30\%$ for $\alpha_{lm} \leq 0.25$, which is a value much greater than the value of p_{obj} by which authors' research is motivated.

Mission (re)planning execution time

Another criterion of performance was the total mission planning and replanning Central processing unit (CPU) time. In Fig.3.9dare given mean execution times of missions for various α_{lm} and p_{obj} values which were used in the statistical analysis benchmarks. The "sawtooth" pattern visible in Fig.3.9dis due to α_{lm} changing its value. It can be noted that WRBNF-CPP method's execution times of the order of $10s$, which can still be considered online, taking into account that the vehicle will need to replan its path while it changes LM lanes at least $50m$ apart which takes around $30 - 50s$ to traverse at the given surge speed of $1.5m/s$.
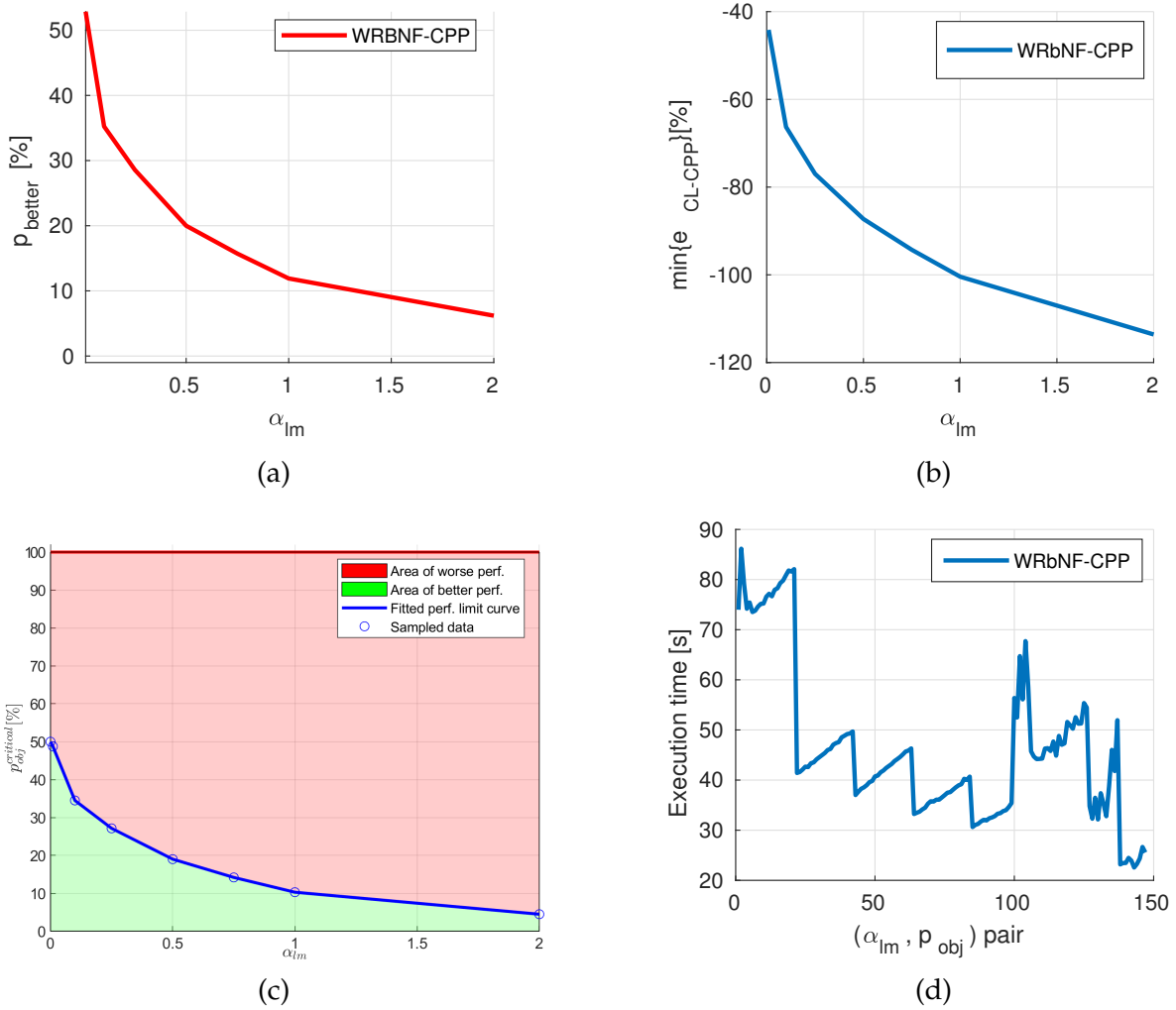
Figure 3.9: (a) Percentage $p_{better}[\%]$ of all simulations ran for the WRBNF-CPP method, for each α_{lm}, in which performance of that method is better than the CL-CPP algorithm. (b) Worst case values of $e_{CL-CPP}^{WRbNF-CPP}$ for each varied value of α_{lm}. (c) Comparison of the percentages $p_{critical}$ of LM segments containing the interesting object(s) needed for the performance of the WRBNF-CPP method to be the same as the CL-CPP method. (d) Mean execution times of the WRBNF-CPP method for the whole mission for each pair of varied values (α_{lm}, p_{obj}).

System engineer's interpretation of the results

The results shown in Fig.3.8a are important from a system engineer's perspective, since they simplify the choice of the mission parameters L, based on the chosen side-scan sonar range $W = w_{sss}$, and an estimate of possible survey mission average duration reduction for an a-priori known value of α_{lm}, and a rough estimate of p_{obj}. It is also possible to extrapolate the relative improvement performance function $e_{CL-CPP}^{WRbNF-CPP}(\alpha_{lm}, p_{obj})$ based on the sampled mean performance values of the WRBNF-CPP algorithm shown in Fig.3.8a.

3.5.9 Conclusion

In this section an online side-scan sonar data-driven complete coverage path planning algorithm for unknown large-scale marine areas has been presented. The essence of the above-mentioned algorithm is the goal to overlap neighboring sonar ranges only if they contain high information gain defined specifically for the current mission, in order to remove sonic shadows around the interesting objects at the sea floor. It is designed bearing in mind that in most exploration and survey missions, LM segments are significantly longer than wider and that only a small part of the coverage area is interesting for the mission.

The proposed WRBNF-CPP uses path planning to diverge from the initial CPP solution LM pattern in the case that it detects interesting objects in the previous LM lane's side-scan sonar image. Upper and lower algorithm performance bounds are estimated analytically in terms of the coverage path and are validated through extensive mission parameters variation simulations. Simulation results show significantly shorter coverage paths obtained by the proposed CPP algorithm compared to the CL-CPP approach, which in a limiting case, results in a twice shorter coverage path.

The next goal was to further improve WRBNF-CPP algorithm by improving its lower performance bound and also to minimize its coverage path length even further. Also, a future research goal is to extend this general survey approach to planning highly detailed sonar images of interesting locations at lower altitudes, and going even deeper to get visual data of the interesting locations. Such a multi-level 3D mission planner would provide at least equally informative, or even more informative side-scan sonar images, as well as visual data of the interesting locations, in a completely autonomous manner. Processing of the data could render 3D model from visual data by using photogrammetry methods, or even merging of the side-scan sonar data with the visual data, thus enabling further insight into many great wonders of the underwater world still waiting to be discovered.

3.6 Basic Accordion Coverage Path Planning Algorithm

In this Section, the proposed coverage algorithm which we named Basic Accordion Coverage Path Planning Algorithm (BA-CPP) is described. It is a solution to the problem defined in Subsection 1.2. The name "accordion" was inspired by the way this algorithm rearranges the LM pattern, expands and contracts it in some parts, just as the accordion bellows do while accordion is played. The idea for designing this algorithm stemmed from the ever-present drive to improve the performance of CPP algorithms.

WRBNF-CPP presented in the previous section has much better performance than the classical overlap-all-sonar-swaths CL-CPP algorithm when, as assumed, the interesting objects are very sparse, and LM lines' length is much greater than LM lane width, which is mostly the case in side-scan sonar exploration missions. However, for increased values of these mission parameters, the performance of the WRBNF-CPP algorithm deteriorated and was in some cases even worse than CL-CPP algorithm w.r.t. the coverage path performance metric. Also, its computation time is growing exponentially with the increase in size of cost map, i.e. the area of each individual LM lane. This section gives a description and performance analysis of a CPP algorithm based on heuristics in an attempt to further improve and speed up side-scan sonar data-based CPP algorithms. These results are published in [39].

3.6.1 Behavior of the proposed algorithm

The initial solution of the BA-CPP algorithm is an LM pattern with lanes twice as wide as the sonar range. This is, of course, a good solution only if there is nothing of interest for the mission in the area to be covered. In case that the vehicle detects some interesting object(s) in its sonar data while traversing the current LM line, the coverage path planning algorithm should replan the rest of the mission in such a way to sonify the interesting object(s) from the opposite side, if that has not been already done. Also, it should again (optimistically) assume that during the rest of the mission it will not encounter any interesting objects, and thus it will again generate the "stretched" LM pattern for the remaining part of the mission. Moreover, in the limiting case when the sea floor area of interest is densely covered with interesting objects, the proposed coverage path planning algorithm should behave as the CL-CPP.

Since the proposed CPP solution is a regular pattern, at least in the initial moment of the mission, elementary periods of the CL-CPP and the BA-CPP approach are given in Fig. 3.10h and 3.10a, respectively. This is needed to analytically predict the behavior of the CPP algorithm proposed in this section on a spatial period of width $4W$, which ends with a horizontal line. This way every LM pattern spatial period starts with a

vertical line and is of the same width. Heuristics behind the area covering of CL-CPP, ensure that complete coverage is achieved by the end of the mission, under the assumptions stated in Section 1.2. This is why it is not insisted that the algorithm achieves a full coverage of the current pattern spatial period, but instead to achieve globally complete coverage in terms of acquiring all informative data present within the whole coverage area.

Algorithm 1 Basic Accordion Coverage Path Planner

  1: WPs = add_case_0(pose,W,α_{lm},L,A)
  2: $k = 0$
  3: **while** $x_{auv} \leq A$ **do**
  4:      pose = WPs(k);
  5:      pose = move_to_next_wp(pose, WPs)
  6:      C = cost_map(*sss_image(WPs(k):WPs(k+1))*)
  7:      **if** contains_int_obj(C) **then**
  8:          **if** (obj_to_the_right&right_not_resonified) **then**
  9:             WPs($k+1:end$) = [ ]
10:             **if** (direction = 'down') **then**
11:                WPs.add_case_1(pose,W,L,A)
12:                WPs.add_case_0(WPs(end),W,L,A)
13:             **else if** (direction = 'up') **then**
14:                WPs.add_case_2(pose,W,L,A)
15:                WPs.add_case_0(WPs(end),W,L,A)
16:          **else if** (obj_to_the_left&left_not_resonified) **then**
17:             WPs($k+1:end$) = [ ]
18:             **if** (direction = 'down') **then**
19:                WPs.add_case_3(pose,W,L,A)
20:                WPs.add_case_0(WPs(end),W,L,A)
21:             **else if** (direction = 'up') **then**
22:                WPs.add_case_4(pose,W,L,A)
23:                WPs.add_case_0(WPs(end),W,L,A)
24:          **else if** (obj_to_the_right&obj_to_the_left&
25: &right_not_resonified&left_not_resonified) **then**
26:             WPs($k+1:end$) = [ ]
27:             **if** (direction = 'down') **then**
28:             WPs.add_case_5(pose,W,L,A)
29:             WPs.add_case_0(WPs(end),W,L,A)
30:             **else if** (direction = 'up') **then**
31:             WPs.add_case_6(pose,W,L,A)
32:             WPs.add_case_0(WPs(end),W,L,A)
33:      $k \leftarrow k + 1$

The characteristic behavior of the proposed algorithm in specific cases of interesting objects' distribution is formally described in Algorithm 1, and graphically represented in Fig. 3.10 (Case 0). If the vehicle encounters no interesting objects in the sonar image

of the current LM line, which is the best-case scenario, it will follow the initial LM pattern, see Fig.3.10a, line 1 in Algorithm1. Method *add_case_0(pose, W, L, A)* generates LM waypoints (WPs) from the starting position *pose*, with LM lanes of width $2W$ and length L until LM lanes reach coverage area's width A.

If the vehicle detects something interesting to its right in the cost map while moving "downwards" along the line 3 as in Fig.3.10b(Case 1), it will generate new waypoints in order to perform a loopback to its right side (LM lines 4, 5, and 6), and thus sonify interesting objects' sonic shadow(s) from the opposite side, given in lines 11 and 12 in Algorithm1.

It is important to note that methods *add_case_1-6(pose, W, L, A)* generate LM waypoints (WPs) from the starting position *pose*, with LM lanes of width W and length L as graphically represented in Fig.3.10for each corresponding case analyzed. This replans the coverage path based on local sonar information gain. Calling the method *add_case_0(pose, W, L, A)* after this step generates LM lanes $2W$ apart until coverage area's edge, assuming no further interesting object detection.

If there is something in the current sonar range to the right of the vehicle, as is the case while the vehicle moves "upwards" along line 1 in Fig.3.10c(Case 2), the BA-CPP algorithm will generate new waypoints of LM line 3, W meters to the right of the LM line 1, to ensure sonar shadows sonification from the opposite side, and continue generating $2W$-wide LM pattern, see lines 14 and 15 in Algorithm1.

In case that the vehicle detects interesting objects to its left side, e.g. while traversing "downwards" along the LM line 3 in Fig.3.10d(Case 3), BA-CPP will generate a new LM line 5, which is W meters to the left of the vehicle. After this line, it sets the line 6, again assuming no further LM lanes containing interesting objects, see lines 19 and 20 in Algorithm1. If on the other hand, the vehicle encounters an interesting object to its left side while traversing "upwards" along the LM line 3 in Fig.3.10e(Case 4), BA-CPP will generate a new loopback LM line 5, which is W meters to the left of the vehicle, and add the line 6, again assuming no further LM lanes containing interesting objects, see lines 22 and 23 in Algorithm1.

If the vehicle detects interesting objects both to its left and right side, e.g. while following LM line 3 "downwards" in Fig.3.10f(Case 5), it will first do a loopback maneuver to its right side (LM lines 4, 5, and 6), and follow a new LM line 7, which is W meters to the right of the LM line 3. After this line, it sets the LM line 8, again assuming no further LM lanes containing interesting objects. This is given in pseudocode in lines 28 and 29 in Algorithm1. In the opposite case, when the vehicle moves "up" the LM line 3 as in Fig.3.10g(Case 6), BA-CPP will generate a loopback to the left of the vehicle (LM lines 4, 5, and 6), and line 7 in order to sonify acoustic shadows of the objects to

the right of its current LM line 3. This process is described by pseudocode lines 31 and 32 in Algorithm1.

In the limiting case, in which each LM lane contains interesting objects, the BA-CPP algorithm behaves as the CL-CPP, see Fig.3.10h(Case 7). The worst-case scenario occurs when the vehicle encounters the characteristic layout of LM lanes containing interesting objects, as shown in Fig.3.10i(Case 8). The coverage path length for the current LM spatial period stays the same as in Cases 5 and 6, but in this case, the vehicle has to start the next LM spatial period at a line which is W away from line 7, making its cumulative coverage path significantly longer. Table3.2shows path lengths for all the above-mentioned cases in one LM pattern spatial period, which are based on graphs depicted in Fig.3.10a-3.10i.

3.6.2 Analysis of the upper and lower performance bounds

In this subsection the best- and the worst-case scenarios for the BA-CPP performance are analyzed (based on the coverage path lengths of the characteristic cases given in Table3.2) in order to predict the upper and the lower bounds of its performance. Performance measure, denoted by e_{CL-CPP}^{BA-CPP}, is chosen to represent the relative improvement of the BA-CPP generated coverage path length l_{BA-CPP} w.r.t. the CL-CPP path length l_{CL-CPP}. In case that the sea floor area which is being explored is devoid of any interesting objects (see Fig.3.10a), then *the best-case scenario improvement* which the BA-CPP has over the CL-CPP algorithm, i.e. the upper bound on e_{CL-CPP}^{BA-CPP}, is given by:

$$e_{CL-CPP}^{best\,BA-CPP} = \frac{1}{2(1+\alpha_{lm})}100[\%] \tag{3.10}$$

If the parameter $\alpha_{lm} \to 0$, the improvement of the BA-CPP method tends to $e_{CL-CPP}^{BA-CPP} \to E_{best} = 50\%$, ensuring a twice shorter coverage path than in case of the CL-CPP. On the other hand, if $\alpha_{lm} \to \infty$, the relative improvement tends to go to $e_{CL-CPP}^{BA-CPP} \to 0\%$.

In case that the interesting objects are positioned in such a way that the vehicle has do to a maneuver shown in Fig.3.10fin each LM pattern spatial period, then the worst-case scenario improvement which the BA-CPP has over the CL-CPP method, i.e. the lower bound on e_{CL-CPP}^{BA-CPP}, is given by:

$$e_{CL-CPP}^{worst\,BA-CPP} = \frac{-\alpha_{lm}}{2(1+\alpha_{lm})}100[\%] \tag{3.11}$$

This means that for $\alpha_{lm} \to 0$, the BA-CPP is never worse than the CL-CPP, i.e. $e_{CL-CPP}^{BA-CPP} \geq 0\%$. On the other hand, when $\alpha_{lm} \to \infty$, the relative improvement tends to go to $e_{CL-CPP}^{BA-CPP} \to E_{worst} = -50\%$.

(a) Case 0: Best case, sea floor area to be covered is devoid of interesting objects, i.e. the initial solution of the proposed BA-CPP algorithm.

(b) Case 1: something was detected to the right of the vehicle in the cost map, while moving downwards along the line 3

(c) Case 2: something was detected to the right of the vehicle in the cost map, while moving upwards along the line 1

(d) Case 3: something was detected to the left of the vehicle in the cost map, while moving downwards along the line 3

(e) Case 4: something was detected to the left of the vehicle in the cost map, while moving upwards along the line 3

(f) Case 5: something was detected both to the left and to the right of the vehicle in the cost map, while moving downwards along the line 3

(g) Case 6: something was detected both to the left and to the right of the vehicle in the cost map, while moving upwards along the line 3

(h) Case 7: whole area to be covered is fully covered by interesting objects, i.e. CL-CPP pattern.

(i) Case 8: Worst case, added third LM lane in a row to have interesting objects

Figure 3.10: Basic accordion coverage path planning algorithm: Characteristic cases of interesting objects (white ellipses) placement in current LM pattern period left and/or right of the current vehicle path in the cost map. Sonic shadows in side-scan sonar data are denoted by gray areas.

Table 3.2: Path lengths for different specific cases of the BA-CPP algorithm behavior. For reference to the BA-CPP algorithm's special behavior cases see Fig.3.10.

Case No.	Path length of the CL-CPP method	Path length of the BA-CPP method
0	$4L + 4W$	$2L + 4W$
1	$4L + 4W$	$3L + 6W$
2	$4L + 4W$	$3L + 4W$
3	$4L + 4W$	$3L + 4W$
4	$4L + 4W$	$3L + 6W$
5	$4L + 4W$	$4L + 6W$
6	$4L + 4W$	$4L + 6W$
7	$4L + 4W$	$4L + 4W$
8	$4L + 4W$	$4L + 6W$

3.6.3 Simulation results

In order to visually represent the BA-CPP algorithm's behavior in a simulated environment, a few performance examples are given in Fig.3.11, with two values of $\alpha_{lm} \in \{0.01, 0.75\}$, and percentage of sonar ranges containing interesting objects $p_{obj} \in \{0, 25, 75, 100\}[\%]$. It is clear that, when $\alpha_{lm} = 0.01$, the BA-CPP algorithm produces, in terms of mean values, coverage paths which are at best approximately 50% shorter than CL-CPP pattern (see Fig.3.11a), and at worst a coverage path of equal length (see Fig.3.11d). An example for $\alpha_{lm} = 0.75$ and $p_{obj} = 75\%$ is given in Fig.3.11c to show when BA-CPP performance becomes worse than CL-CPP by 12%.

The better BA-CPP performance is, the more CL-CPP (magenta) lines are left uncovered, see Fig.3.11a and Fig.3.11b. Worsened BA-CPP performance are equivalent to more CL-CPP (magenta) lines being overlaid by BA-CPP (dashed cyan) lines, due to loopback maneuvers shown in Fig.3.10b and 3.10d, here notable in Fig.3.11c and 3.11d.

3.6.4 Statistical analysis of algorithm performance

In order to gain further insight into the behavior of the BA-CPP algorithms, its performance has been evaluated for various values of mission dependent parameters α_{lm} and p_{obj}. Parameter α_{lm} have been varied as $\alpha_{lm} \in \{0.01, 0.1, 0.25, 0.5, 0.75, 1, 2\}$, while p_{obj} took values from 0% to 100% with a 5% step. For each possible tuple (α_{lm}, p_{obj}),

(a) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 0\%$, $e_{cl-cpp}^{ba-cpp} = 49.5\%$

(b) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 25\%$, $e_{cl-cpp}^{ba-cpp} = 36\%$

(c) $\alpha_{lm} = 0.75$, $C = 2.34km^2$, $p_{obj} = 75\%$, $e_{cl-cpp}^{ba-cpp} = -12\%$

(d) $\alpha_{lm} = 0.01$, $C = 5.5km^2$, $p_{obj} = 100\%$, $e_{cl-cpp}^{ba-cpp} = 0\%$
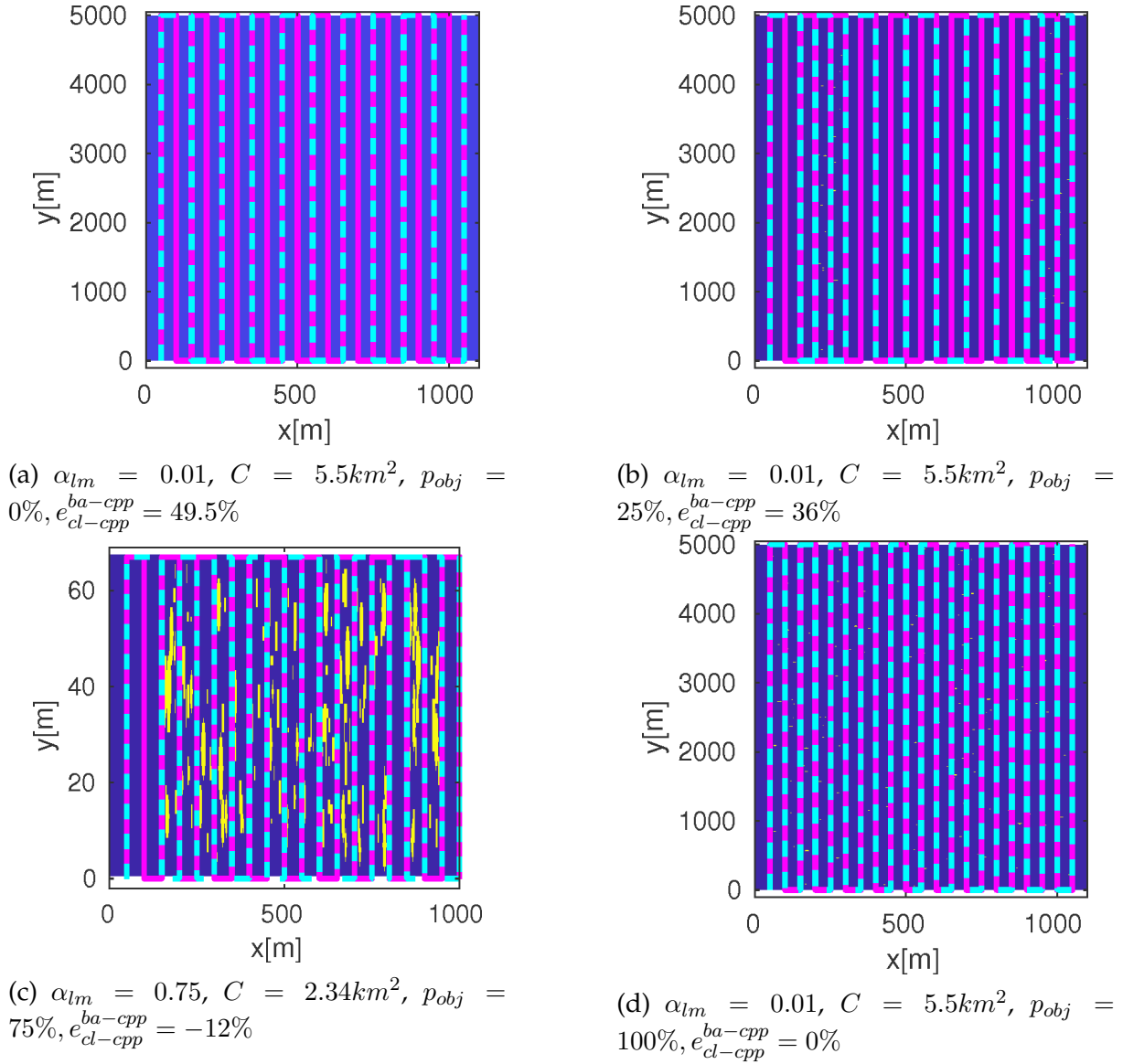
Figure 3.11: Examples of the BA-CPP algorithm performance compared to the CL-CPP pattern, with the visual representation of coverage area and coverage paths. (cyan dashed) path of the vehicle following the BA-CPP solution, (magenta) path of the vehicle following the CL-CPP pattern, (dark blue) low informational gain objects, and (yellow) high information gain objects.

100 tests with appropriate information gain cost maps have been generated, where the interesting objects have been dispersed randomly. Coverage area C had values $C \in \{5.5, 3.8, 3.3, 2.73, 2.34, 2.05, 1.36\}[km^2]$ for each α_{lm}, respectively. Cost maps values were binarised in a way that the value of $c = 0.1$ meant noninteresting areas, while $c = 1$ meant interesting areas. Relative coverage path length improvement of the BA-CPP algorithm over the CL-CPP pattern was averaged over 100 tests per each (α_{lm}, p_{obj}) parameter tuple.

The results of this analysis are shown in Fig.3.12a. It can be noted that the average improvement of the BA-CPP w.r.t. the CL-CPP is most significant in the area defined by low α_{lm}, $\alpha_{lm} \leq 0.25$, with $p_{obj} \leq 20\%$. In these cases, the BA-CPP generates coverage paths which are on average $25 - 50\%$ shorter than the CL-CPP paths. For $\alpha \leq 0.1$, BA-CPP coverage path is mostly shorter than the CL-CPP counterpart.

Best- and worst-case scenario analysis validation

The goal of the aforementioned parameter variation analysis was to validate analytical best- and worst-case performance given in Subsection3.6.2. Aggregated distributions of relative coverage path improvements of the BA-CPP algorithm w.r.t. the CL-CPP for each fixed α_{lm}, and varied p_{obj}, are shown in Fig.3.12b. Each of the boxplots contains the respective improvement values obtained from 2100 simulations ran per each α_{lm} value. It can be noted that the simulation results of the best-case scenarios match the predicted algorithm performance given by Eq.3.10. For each (α_{lm}, p_{obj}) tuple 2 worst-case scenarios, shown in Fig.3.10f, have been generated, and the performance of the BA-CPP algorithm in these cases also matches the predictions given by Eq.3.11.

Fig.3.12cshows the percentage of simulations p_{better}, for each α_{lm}, in which the BA-CPP generated shorter complete coverage paths than the CL-CPP. It can be noted that even for $\alpha_{lm} \leq 0.25$ over 70% of simulations gave better results using the BA-CPP instead of the CL-CPP. This percentage jumps significantly to 90% and even 100% as α_{lm} decreases to values of 0.1, and 0.01, respectively. This fact serves as a good example that shows how the BA-CPP generates shorter complete coverage paths than the CL-CPP in a wide range of parameter α_{lm} values that mostly used in practice.

Limit case performance analysis

To generalize the conclusion about the BA-CPP algorithm performance, sample values of improvement percentages e_{CL-CPP}^{BA-CPP} resulting from the statistical analysis of the BA-CPP performance, have been interpolated by using a polynomial function of the third order of parameters α_{lm} and p_{obj} by using Levenberg-Marquardt method. This function
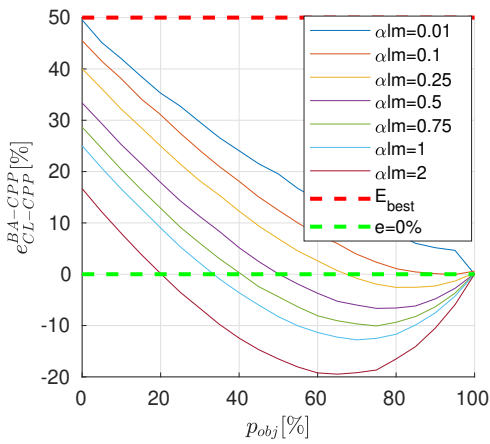
(a) BA-CPP algorithm averaged improvements relative to CL-CPP method.
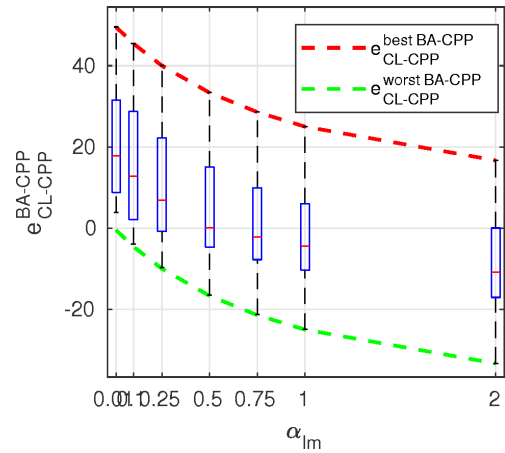
(b) Coverage path lenth improvements distributions of the BA-CPP relative to the CL-CPP.
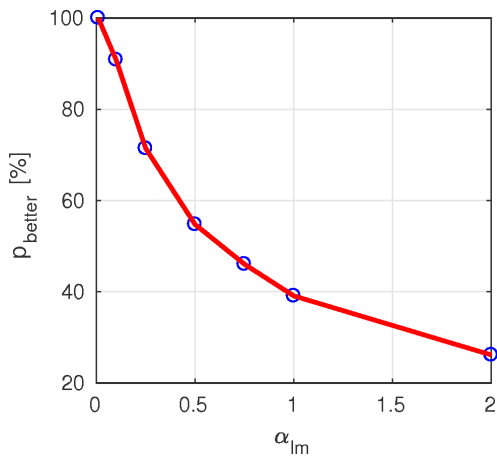
(c) Percentage $p_{better}[\%]$ of 630 simulations ran for each α_{lm} in which performance of the BA-CPP was better than the CL-CPP algorithm.
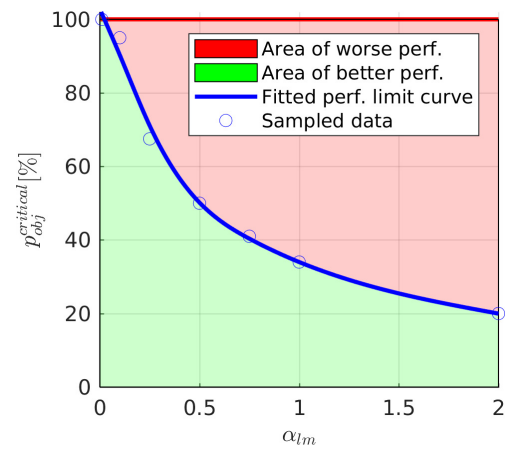
(d) Critical percentage of coverage area populated with interesting objects $p_{obj}^{critical}$ for which performance of the BA-CPP deteriorates to performance of the CL-CPP.

Figure 3.12: BA-CPP algorithm performance analysis.

is given by the following equation:

$$e_{CL-CPP}^{BA-CPP}(\alpha_{lm}, p_{obj}) = \sum_{i=0}^{3} \sum_{j=0}^{3} k_{ij} \alpha_{lm}^{i} p_{obj}^{j} \tag{3.12}$$

where $k_{00} = 50.2$, $k_{10} = -52.42$, $k_{01} = -0.6427$, $k_{20} = 34.86$, $k_{11} = -0.09411$, $k_{02} = -0.002369$, $k_{30} = -8.262$, $k_{21} = -0.05325$, $k_{12} = 0.00327$, and $k_{03} = 4.38e-05$ are nonzero coefficients estimated within 95% confidence bounds. This result is important from a system engineer's perspective, since it allows an estimate of possible survey mission average speed-up based on an a-priori known value of α_{lm}, and a rough estimate of p_{obj}. Conversely, based on a desired mission speed-up $e_{CL-CPP}^{BA-CPP}[\%]$ and an expected value of p_{obj}, it is possible to get the value of coverage mission design parameter α_{lm}, and divide coverage area accordingly if needed.

Curve $e_{CL-CPP}^{BA-CPP}(\alpha_{lm}, p_{obj}) = 0 = p_{obj}^{critical}(\alpha_{lm})$ is shown in Fig.3.12d. This curve represents the limit case in which performance of the BA-CPP algorithm cannot be guaranteed to be on average better than performance of the CL-CPP. It can also be generated based on the graph in Fig.3.12a as a curve connecting zeros of all the curves as a function of belonging α_{lm} values.

The main conclusion that needs to be stressed is that even when $\alpha_{lm} = 1$, the coverage area can contain even 35% of LM lanes populated with interesting objects in order for the BA-CPP algorithm to have on average the same performance as the CL-CPP method. For $\alpha_{lm} \leq 0.25$, the percentage of LM lanes of interest needs to be more than 70% for the BA-CPP to become equally inefficient as the CL-CPP. Since it is assumed that $p_{obj} \ll 50\%$, the BA-CPP is thus expected to be better than the CL-CPP in almost any case of large-scale underwater area scarcely populated with interesting objects.

3.6.5 Conclusion

The BA-CPP is an online side-scan sonar data-driven complete coverage path planning algorithm for unknown large-scale marine areas. It is designed bearing in mind that in most exploration and survey missions, LM segments are significantly longer than wider, and that only a small part of the coverage area is interesting for the survey mission. The BA-CPP algorithm overlaps neighboring side-scan sonar swaths only if they contain high information gain defined specifically for the current mission. The proposed algorithm replans the rest of the coverage mission taking estimated local sonar data information gain into consideration. Algorithm's upper and lower performance bounds are estimated analytically.

Its performance is tested through extensive mission parameters variation simula-

tions, which validate the modeled performance bounds. Simulation results show significantly shorter coverage paths obtained by the BA-CPP algorithm compared to the CL-CPP approach, which in a limiting case, results in a twice shorter coverage path.

The next goal was to improve the BA-CPP algorithm by tightening its lower bounds of performance when it becomes worse than the CL-CPP algorithm and also to minimize coverage path length even further.

3.7 Extended Accordion Coverage Path Planning Algorithm

The proposed coverage algorithm which we named Extended Accordion Coverage Path Planning (EXTA-CPP) is described in this section. The idea for designing this algorithm stemmed from the ever-present drive to improve the performance of CPP algorithms. BA-CPP presented in the previous section has much better performance than the classical overlap-all-sonar-swaths CL-CPP algorithm when, as assumed, the interesting objects are very sparse, and LM lines' length is much greater than LM lane width, which is mostly the case in side-scan sonar exploration missions. BA-CPP algorithm's performance was even better than WRBNF-CPP algorithm's performance on a wider range of mission parameters. However, for increased values of these mission parameters, the performance of the BA-CPP algorithm deteriorated and was in some cases even 10% worse than CL-CPP algorithm w.r.t. the coverage path performance metric.

This section gives a description and performance analysis of a CPP algorithm that is based on heuristics of BA-CPP algorithm for initial adaptive mission (re)planning w.r.t. the available side-scan sonar data. At some point, as seen in the previous section, this adaptive behavior can lead to worse performance compared to nonadaptive CL-CPP algorithm. Exactly this is the main idea behind extending the BA-CPP algorithm into the EXTA-CPP algorithm, to start the mission with adaptive coverage planning, but when the percentage of LM lanes containing interesting objects and the LM lane width-to-length ratio reach critical values for which BA-CPP has the same performance as CL-CPP - then switch to nonadaptive CL-CPP coverage planning in order to narrow down the lower performance limit w.r.t. the coverage path length.

3.7.1 Behavior of the proposed algorithm

The main drawback of the results of statistical analysis of the BA-CPP algorithm [39] is that its worst-case mean improvements in coverage path w.r.t. the CL-CPP method

are negative and even below -10%.

This problem has been successfully alleviated as much as possible by the Extended Accordion CPP (EXTA-CPP) algorithm presented in this Subsection. In essence, the goal was to extend the BA-CPP algorithm so that it starts the CPP mission behaving the same way as the BA-CPP algorithm (described in detail in [39]), but then switch to behaving the same as the CL-CPP method as soon as it estimates that the percentage p_{obj} of LM lanes containing interesting objects has reached the critical value $p_{obj} = p_{critical}(\alpha_{lm})$. The value of $p_{critical}(\alpha_{lm})$ represents a function given by:

$$p_{critical}(\alpha_{lm}) = p_{obj} \ni e_{cl-cpp}^{ba-cpp}(\alpha_{lm}, p_{obj}) \triangleq 0. \tag{3.13}$$

Function $e_{cl-cpp}^{ba-cpp}(\alpha_{lm}, p_{obj})$ represents the relative coverage path/duration improvement of the BA-CPP algorithm w.r.t. the CL-CPP method, which depends on mission parameters α_{lm} and p_{obj}, and it can be extrapolated based on results shown in Fig. 3.12d, and values $p_{critical}(\alpha_{lm})$ actually represent its zeros for corresponding values of the mission parameter α_{lm}. Coverage path lengths of the EXTA-CPP algorithm are thus in the worst case equally long as the BA-CPP coverage paths.

3.7.2 Analysis of the upper and lower performance bounds

Since EXTA-CPP algorithm has been designed to improve only BA-CPP algorithm's worse performance w.r.t. the CL-CPP method, its *best-case scenario performance* improvement $e_{cl-cpp}^{best\,exsta-cpp}$ is equal to the best-case scenario performance improvement of the BA-CPP algorithm, given by Eq. (3.14).

$$e_{cl-cpp}^{best\,exsta-cpp} = e_{CL-CPP}^{best\,BA-CPP} = \frac{1}{2(1+\alpha_{lm})}100[\%] \tag{3.14}$$

It can be deduced that in *the worst-case scenario*, the improvement of the EXTA-CPP algorithm w.r.t. the CL-CPP, denoted by $e_{cl-cpp}^{worst\,exta-cpp}$ will be better than the one of the BA-CPP algorithm $e_{cl-cpp}^{worst\,ba-cpp}$ given by Eq. (3.11), i.e.

$$e_{cl-cpp}^{worst\,exta-cpp} \geq \left(e_{cl-cpp}^{worst\,ba-cpp} = \frac{-\alpha_{lm}}{2(1+\alpha_{lm})}100[\%] \right). \tag{3.15}$$

This is due to the fact that EXTA-CPP algorithm avoids the negative effect of the BA-CPP algorithm's loop-back maneuvers in the worst-case scenario, and instead of that plans the rest of the mission in a nonadaptive fashion as the CL-CPP method does, without any further loop-backs. Also, the results show that $e_{cl-cpp}^{exta-cpp} \geq -5\%, \forall (\alpha_{lm}, p_{obj})$.

3.7.3 Statistical analysis of simulation results

In order to gain further insight into the performance of the proposed EXTA-CPP algorithm, extensive parameter variation simulations have been conducted with the same settings as described in [39] and [39]. Parameter α_{lm} was varied as $\alpha_{lm} \in \{0.01, 0.1, 0.25, 0.5, 0.75, 1, 2\}$, while p_{obj} took values from 0% to 100% with a 5% step. For each possible tuple (α_{lm}, p_{obj}), 100 tests with appropriate information gain cost maps have been generated, where the interesting objects have been dispersed randomly. Coverage area C had values $C \in \{5.5, 3.8, 3.3, 2.73, 2.34, 2.05, 1.36\}[km^2]$ for each α_{lm}, respectively. Relative coverage path length improvement of the presented algorithms over the CL-CPP pattern was averaged over 100 tests per each (α_{lm}, p_{obj}) parameter tuple.

The results of this analysis are shown in Fig.3.13a. It can be noted that the average improvement of the EXTA-CPP w.r.t. the CL-CPP is the most significant in the area defined by low α_{lm}, $\alpha_{lm} \leq 0.25$, with $p_{obj} \leq 20\%$. In these cases the EXTA-CPP generates coverage paths which are on average $25 - 50\%$ shorter than the CL-CPP paths. For $\alpha \leq 0.1$, EXTA-CPP coverage path is mostly shorter than the CL-CPP counterpart.

Comparing Fig.3.13aand Fig. 2a from [39] it can be noted that the inequality $min\{e_{cl-cpp}^{exta-cpp}\} \geq min\{e_{cl-cpp}^{ba-cpp}\}$ holds for every (α_{lm}, p_{obj}) pair, which is in accordance with Eq. (3.11).

Best- and worst-case scenario analysis validation

Distributions of relative coverage path improvements of the EXTA-CPP algorithm w.r.t. the CL-CPP for each fixed α_{lm}, and varied p_{obj}, are shown in Fig.3.13b. Each of the boxplots contains the respective improvement values obtained from 630 simulations ran per each α_{lm} value. It can be noted that the simulation results of the best-case scenarios match the predicted algorithm's best performance (given in Subsection3.7.2). The worst-case performance shown in Fig.3.13balso match the predictions given by Eq.3.11. It is notable that EXTA-CPP algorithm's design makes its performance always above the worst-case performance of the BA-CPP method, as was predicted by Eq. (3.11).

Limit case performance analysis

Curve $e_{CL-CPP}^{EXTA-CPP}(\alpha_{lm}, p_{obj}) = 0 = p_{obj}^{critical}(\alpha_{lm})$ is shown in Fig.3.13c. This curve represents the limit case in which performance of the EXTA-CPP algorithm cannot be guaranteed to be on average better than performance of the CL-CPP. The main conclusion that needs to be stressed is that even when $\alpha_{lm} = 1$, the coverage area can contain even

(a)

(b)

(c)
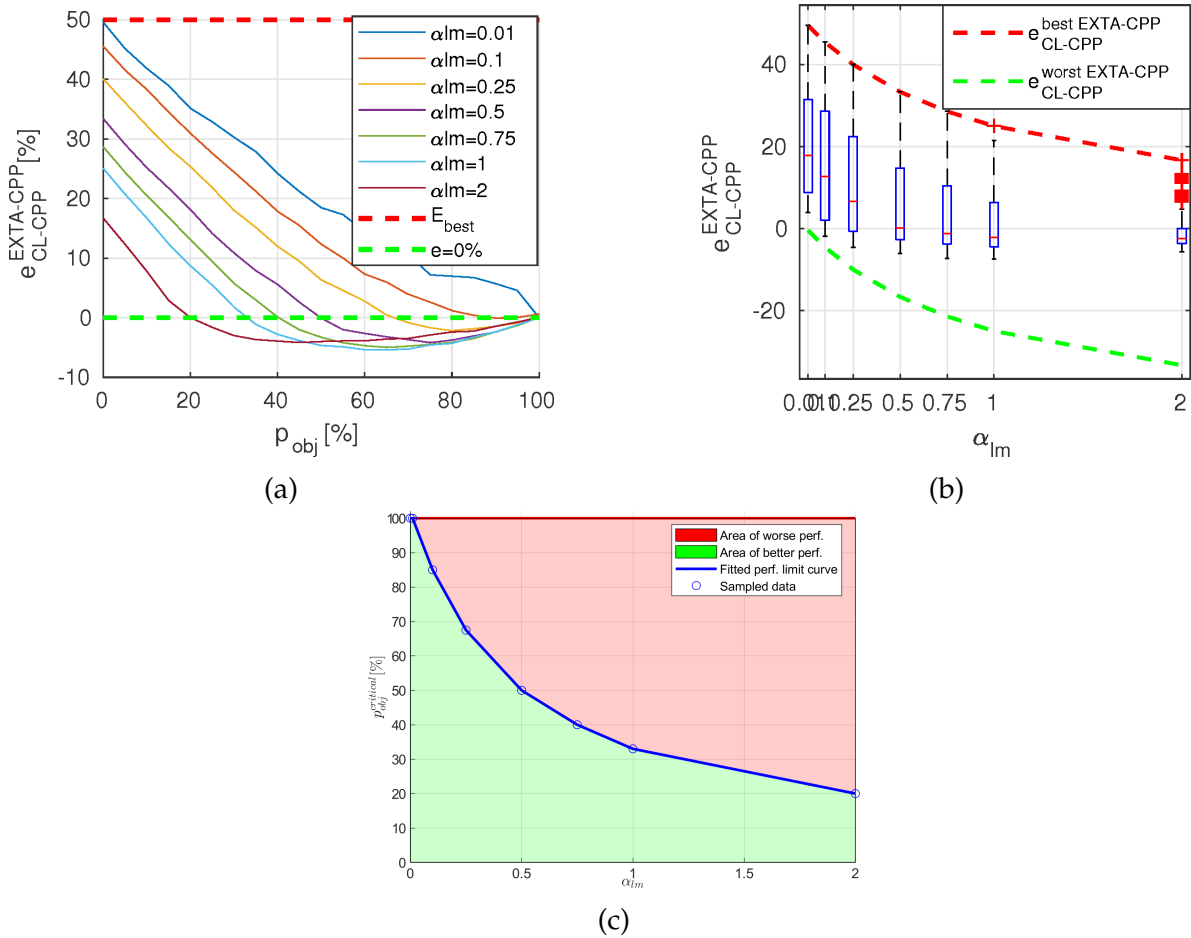
Figure 3.13: (a) Analysis of parameter variation simulations for EXTA-CPP method. (b) Upper and lower bounds of EXTA-CPP algorithm performance validation. (c) Limit case performance of EXTA-CPP performance w.r.t. the CL-CPP algorithm.

30% of LM lanes populated with interesting objects in order for the EXTA-CPP algorithm to have on average the same performance as the CL-CPP method. For $\alpha_{lm} \leq 0.25$, the percentage of LM lanes of interest needs to be more than 67% for the EXTA-CPP to become equally inefficient as the CL-CPP. Since it is assumed that $p_{obj} \ll 50\%$, the EXTA-CPP is thus expected to be better than the CL-CPP in almost any case of large-scale underwater area scarcely populated with interesting objects.

3.8 Optimized Accordion Coverage Path Planning Algorithm

The main drawback of the EXTA-CPP algorithm is the fact that its performance deteriorates at approximately the same rate as the BA-CPP algorithm. The reason for this behavior is that the EXTA-CPP, as well as any other online CPP algorithms, does not know at what point the percentage of LM lanes containing interesting objects p_{obj} will reach its critical value, after which accordion-based CPP methods show worse performance than the CL-CPP method.

The results shown in Fig.3.13agave rise to another idea that an Optimized Accordion-based Coverage Path Planning (OPTA-CPP) algorithm could be designed, which would replan its mission taking into account not only if the vehicle detects some interesting objects to the left and/or right of it in the current LM lane's side-scan sonar data, but also taking into account the range of dispersion of these objects along the LM lane.

3.8.1 Behavior of the proposed algorithm

The proposed OPTA-CPP algorithm starts off coverage mission behaving the same way as the BA-CPP algorithm, but with shorter loopback paths in case that the vehicle detects some interesting object(s) left and/or right of its current LM path. Characteristic 9 cases of decision-making based on the current LM spatial pattern period containing arbitrarily positioned interesting objects is described in detail in [39] and for brevity is not repeated here. Cases 0, 1, 3, 4, 5, and 6 given in Fig. 1 in [39] are here analyzed as cases 0, 1, 2, 3 in Fig.3.14.

Instead of traversing the whole leg length L when an interesting object is detected, the vehicle replans its coverage path to traverse only a portion of that length equal to the range of interesting objects' dispersion $100k[\%], k \in [0, 1]$ in the current LM lane, see Fig.3.14. Also, the vehicle in these cases would not, generally speaking, move along horizontal lines of lawnmower segment width W length, but instead, it would move diagonally towards the portion of the newly generated LM line, which is $2W$ away

from the currently followed line, see Fig.3.14. All characteristic cases of algorithm behavior are given in Fig.3.14. Based on everything previously stated, and based on the graphical representation of the above-mentioned characteristic cases in Fig.3.14, coverage path lengths of the OPTA-CPP algorithm in such cases are evidently always shorter than the ones that the BA-CPP approach generates, and thus $e_{cl-cpp}^{opta-cpp} \geq e_{cl-cpp}^{ba-cpp}$.

The reason why BA-CPP has been chosen for the above-mentioned coverage path optimization and not the EXTA-CPP algorithm (which has been shown to have much tighter lower performance bounds) is because otherwise $p_{critical}$ would have to be estimated for each $k \in [0,1]$. This is of course possible to do, and is a part of our future work.

3.8.2 Analysis of the upper and lower performance bounds

Best-case scenario performance improvement $e_{cl-cpp}^{best\,opta-cpp}$ of the OPTA-CPP algorithm could be described the same way as for the BA-CPP algorithm (see Fig.3.14a), i.e. $e_{cl-cpp}^{best\,opta-cpp} = e_{cl-cpp}^{best\,bA-cpp}$.

Worst-case scenario for one LM pattern spatial period is shown in Fig.3.14d. Two types of worst-case scenarios can be defined for the OPTA-CPP algorithm. Let us assume that interesting objects shown in Fig.3.14d are located in the range of first $[k\%]$ of each LM lane length. Now the path length of the OPTA-CPP algorithm can be parametrized by the parameter k as well:

$$l_{opta-cpp} = 2L + 2kL + 6W = 2L(1 + k + 3\alpha_{lm}). \tag{3.16}$$

With CL-CPP method's fixed coverage path length per one spatial period (see Fig.3.2a) $l_{cl-cpp} = 4L + 4W$, the relative improvement of the OPTA-CPP algorithm w.r.t. the CL-CPP method in the worst case is given by:

$$e_{cl-cpp}^{worst\,opta-cpp} = \frac{1 - k - \alpha_{lm}}{2(1 + \alpha_{lm})} \geq e_{cl-cpp}^{worst\,ba-cpp}. \tag{3.17}$$

In the *worst-case scenario* in which all the interesting objects are dispersed along the whole length of all LM lanes, i.e. $k \to 1$, and thus we obtain:

$$e_{cl-cpp}^{worst\,opta-cp} \to \frac{-\alpha_{lm}}{2(1 + \alpha_{lm})} = e_{cl-cpp}^{worst\,ba-cpp}. \tag{3.18}$$

On the other hand, in the *worst-case scenario* in which all the interesting objects are dispersed along an infinitesimally short part of each LM lane, at its bottom or top, i.e.

(a) Case 0: best case - sea floor is completely barren

(b) Case 1: something was detected in the swath to the left of the vehicle in the cost map

(c) Case 2: nsomething was detected in the swath to the right of the vehicle in the cost map

(d) Case 3: something was detected in the swaths both to the left and right of the vehicle in the cost map

Figure 3.14: Optimized accordion coverage path planning algorithm: Characteristic cases for specific placement of interesting objects left or right of the current vehicle path in the cost map.

$k \to 0$, and thus we obtain:

$$e_{cl-cpp}^{worst\ opta-cpp} \to \frac{1 - \alpha_{lm}}{2(1 + \alpha_{lm})} > e_{cl-cpp}^{worst\ ba-cpp}. \tag{3.19}$$

Based on Eq. (3.18) and (3.19), it can be concluded that the theoretical lower bound on OPTA-CPP algorithm's performance is the BA-CPP algorithm's worst-case performance. As the range of interesting objects' dispersion along the LM lanes' length decreases, i.e. $k \to 0$, so does the OPTA-CPP algorithm become increasingly better than the BA-CPP algorithm.

3.8.3 Statistical analysis of simulation results

In order to gain further insight into the performance of the proposed OPTA-CPP algorithm, extensive parameter variation simulations have been conducted with the same settings as described in Subsection 3.7.3. The results of this analysis for interesting objects dispersed in the lower $k = 10\%$ of LM lanes are shown in Fig. 3.15a. It can be noted that the average improvement of the OPTA-CPP w.r.t. the CL-CPP is the most significant in the area defined by low α_{lm}, $\alpha_{lm} \leq 0.5$, with $p_{obj} \leq 20\%$. In these cases the OPTA-CPP generates coverage paths which are on average $25 - 50\%$ shorter than the CL-CPP paths. For $\alpha \leq 0.5$, OPTA-CPP coverage path is on average always shorter than the CL-CPP counterpart.

Comparing Fig. 3.13a, 3.15a, and Fig. 2a from [39], it can be noted that the following inequalities hold: $min\{e_{cl-cpp}^{opta-cpp}(k = 0.1)\} \geq min\{e_{cl-cpp}^{ba-cpp}\}$ and $min\{e_{cl-cpp}^{opta-cpp}(k = 0.1)\} \geq min\{e_{cl-cpp}^{exta-cpp}\}$, for every (α_{lm}, p_{obj}) pair, which is in accordance with Eq. (3.18) and (3.19).

Best- and worst-case scenario analysis validation

Distributions of relative coverage path improvements of the OPTA-CPP for k=10% algorithm w.r.t. the CL-CPP for each fixed α_{lm}, and varied p_{obj}, are shown in Fig. 3.15b. Each of the boxplots contains the respective improvement values obtained from 630 simulations ran per each α_{lm} value. It can be noted that the simulation results of the best-case scenarios match the predicted algorithm's best performance (given in Subsection 3.8.2). Moreover, OPTA-CPP's worst-case performance match the predictions given by Eq. (3.18). It is notable that OPTA-CPP algorithm's design makes its worst-case performance always better than the worst-case performance of the BA-CPP method for $k = 10\%$, which was predicted by Eq. (3.17).
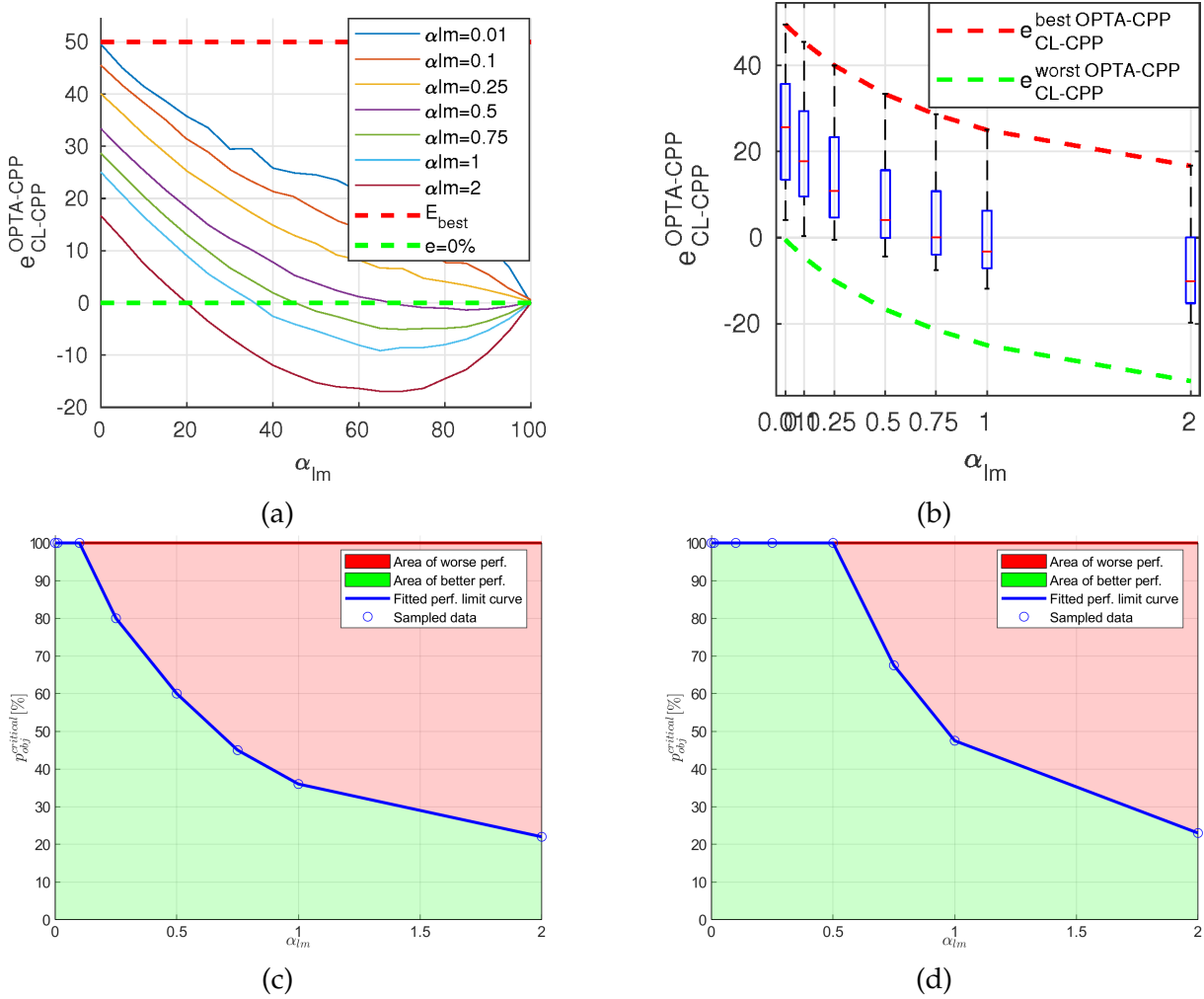
Figure 3.15: (a) Analysis of parameter variation simulations for OPTA-CPP method in case that interesting objects are dispersed in $k = 10\%$ of the LM lane length. (b) Coverage path lenth improvements distributions of the OPTA-CPP relative to the CL-CPP for $k = 10\%$. (c) Limit case performance of OPTA-CPP performance w.r.t. the CL-CPP algorithm for $k = 100\%$. (d) Limit case performance of OPTA-CPP performance w.r.t. the CL-CPP algorithm for $k = 10\%$.

Limit case performance function

Curve $e_{CL-CPP}^{OPTA-CPP}(\alpha_{lm}, p_{obj}) = 0 = p_{obj}^{critical}(\alpha_{lm})$ is shown in Figs.3.15cand3.15dfor cases when interesting objects are dispersed in $k = 100\%$ and $k = 10\%$ of the LM lane length, respectively. This curve represents the limit case in which performance of the OPTA-CPP algorithm cannot be guaranteed to be on average better than performance of the CL-CPP. The main conclusion that needs to be stressed is that even when $\alpha_{lm} = 1$ and $k = 100\%$, see Fig.3.15c, the coverage area can contain over 35% of LM lanes populated with interesting objects in order for the OPTA-CPP algorithm to have on average the same performance as the CL-CPP method. For $\alpha_{lm} \leq 0.25$ and $k = 100\%$, the OPTA-CPP is always better than the CL-CPP. When $k = 10\%$, the average performance compared to the CL-CPP become even better, so much so that the OPTA-CPP is always better than the CL-CPP for α_{lm} going as high as $\alpha_{lm} \leq 0.5$. Since it is assumed that $p_{obj} \ll 50\%$, the OPTA-CPP is thus expected to be better than the CL-CPP in almost any case of large-scale underwater area scarcely populated with interesting objects.

3.9 Comparison of the implemented coverage path planning algorithms

In order to see which of the coverage path planning algorithms proposed in this chapter has the best performance w.r.t. the various performance metrics, in this section this is given in an aggregated way. Performance metrics used to compare different aspects of CPP algorithms' performance were:

1. better mean performance probabilities - giving the dependence of the percentage of simulations w.r.t. in which during extensive parameter variation CPP algorithms presented in this chapter generated shorter coverage paths than the CL-CPP algorithm,

2. worst-case performance analysis - shows the worst-case improvements of the coverage paths generated by all the above-mentioned algorithms w.r.t. the CL-CPP method,

3. limit (critical) case performance analysis - shows in which cases the performance of the CPP algorithms presented in this chapter cannot guarantee to have on average better performance than the CL-CPP method, and

4. mission (re)planning execution time - for coverage planning algorithm complexity analysis and speed vs. improved coverage path trade-off analysis.

3.9.1 Better mean performance probabilities

Fig.3.16a shows the percentage of simulations p_{better}, for each α_{lm}, in which the CPP algorithms presented in this chapter generated shorter coverage paths than the CL-CPP algorithm. WRBNF-CPP approach from [40] has been shown to have better performance than the CL-CPP in 35% of the simulations for $\alpha_{lm} \leq 0.1$. Accordion-based CPP algorithms presented in this chapter have been shown to have better performance in over 90% of the simulations for $\alpha_{lm} \leq 0.1$. BA-CPP and EXTA-CPP algorithms have shown the same performance in this comparison criterion. It is interesting to note how OPTA-CPP for $k = 10\%$ outperforms all three of its predecessors, and is always better than the CL-CPP algorithm for $\alpha_{lm} \leq 0.25$.

3.9.2 Worst-case performance analysis

In order to gain further insight into the less good improvements of the WRBNF-, BA-, EXTA-, and OPTA-CPP algorithms w.r.t. the CL-CPP, Fig.3.16b shows the worst-case improvements of the coverage paths generated by all the above-mentioned algorithms w.r.t. the CL-CPP method, for parameter α_{lm} taking values $\alpha_{lm} \in \{0.01, 0.1, 0.25, 0.5, 0.75, 1, 2\}$.

It is notable that WRBNF-CPP worst-case relative performance is negative for each value of α_{lm}, which is probably caused by the loop-back maneuvers and further divergences of the coverage path from the initially set LM lines. Of all the accordion-based CPP algorithms presented in this chapter, BA-CPP has the lowest worst-case values of the relative improvement w.r.t. the CL-CPP algorithm, which has been expected due to its unoptimized behavior. However, for values $\alpha_{lm} \leq 0.1$, worst-case performance of all implemented CPP algorithms match, and are 0%, which means that in those cases all mentioned CPP algorithms have performance which are not worse than the CL-CPP.

EXTA-CPP algorithm exhibits the mildest worst-case performance for $\alpha_{lm} \geq 0.75$, limiting its worst-case relative improvement w.r.t. the CL-CPP method to around -5%. OPTA-CPP algorithm for dispersion of interesting objects in the $k = 10\%$ of LM lanes' length, shows the best performance in the range $0 \leq \alpha_{lm} \leq 0.5$, which is the target range for the application of the CPP algorithms presented in this chapter. In this range, the OPTA-CPP algorithm never has worse performance than the CL-CPP algorithm.

3.9.3 Limit case performance analysis

Curves $e_{CL-CPP}(\alpha_{lm}, p_{obj}) \triangleq 0 = p_{obj}^{critical}(\alpha_{lm})$ are shown in Fig.3.16c. These curves represent the limit cases in which the performance of the CPP algorithms presented in

this chapter cannot guarantee to have on average better performance than the CL-CPP method.

It can be noted that of all CPP algorithms proposed in this chapter, WRBNF-CPP algorithm has on average equal performance as the CL-CPP algorithm when $p_{obj} \geq 50\%$, which is a value much greater than the value of p_{obj} by which the research for this chapter has been motivated. For $\alpha_{lm} \leq 0.01$, BA-CPP and EXTA-CPP algorithms' performances become equal to the performance of the CL-CPP algorithm only when the coverage area contains interesting objects in each LM lane that the vehicle traverses. When $\alpha_{lm} \leq 0.5$, and $\alpha_{lm} \leq 0.1$, performance of the OPTA-CPP algorithm for dispersion of the interesting objects in $k = 10\%$, and $k = 100\%$ of the LM lanes' length becomes equal to the performance of the CL-CPP algorithm only when $p_{obj} = 100\%$. Furthermore, the OPTA-CPP algorithm's performance drops slower than all the other CPP algorithms' proposed in this chapter, even with $k = 100\%$, which was expected.

Also, when the value of interesting objects' dispersion along the lawnmower length k drops to $k = 10\%$, a significant improvement over the case when $k = 100\%$ is notable.

3.9.4   Mission (re)planning execution time

Another criterion, by which all CPP algorithms presented in this chapter have been compared by, was the total mission planning and replanning CPU time for all conducted simulations. In Fig.3.16d are given mean execution times of missions for various α_{lm} and p_{obj} values which were used in the statistical analysis benchmarks. The "sawtooth" pattern visible in Fig.3.16d is due to α_{lm} changing its value. It can be noted that WRBNF-CPP method's execution times are manyfold longer than the ones of the accordion-based methods, due to its use of dynamical programming for planning. Also, all accordion-based CPP methods show a similar complexity, which was expected, since their design is only slightly different. Moreover, OPTA-CPP method seems to be a bit faster than the other two accordion-based CPP methods, which is most probably due to its shorter path, and less unneeded data readings.

3.10   Conclusion

In this chapter four online side-scan sonar data-driven complete coverage path planning algorithms for unknown large-scale marine areas have been presented. The essence of all algorithms is the goal to overlap neighboring sonar ranges only if they contain high information gain defined specifically for the current mission, in order to sonify interesting objects at the sea floor from both sides. They are designed bearing in mind
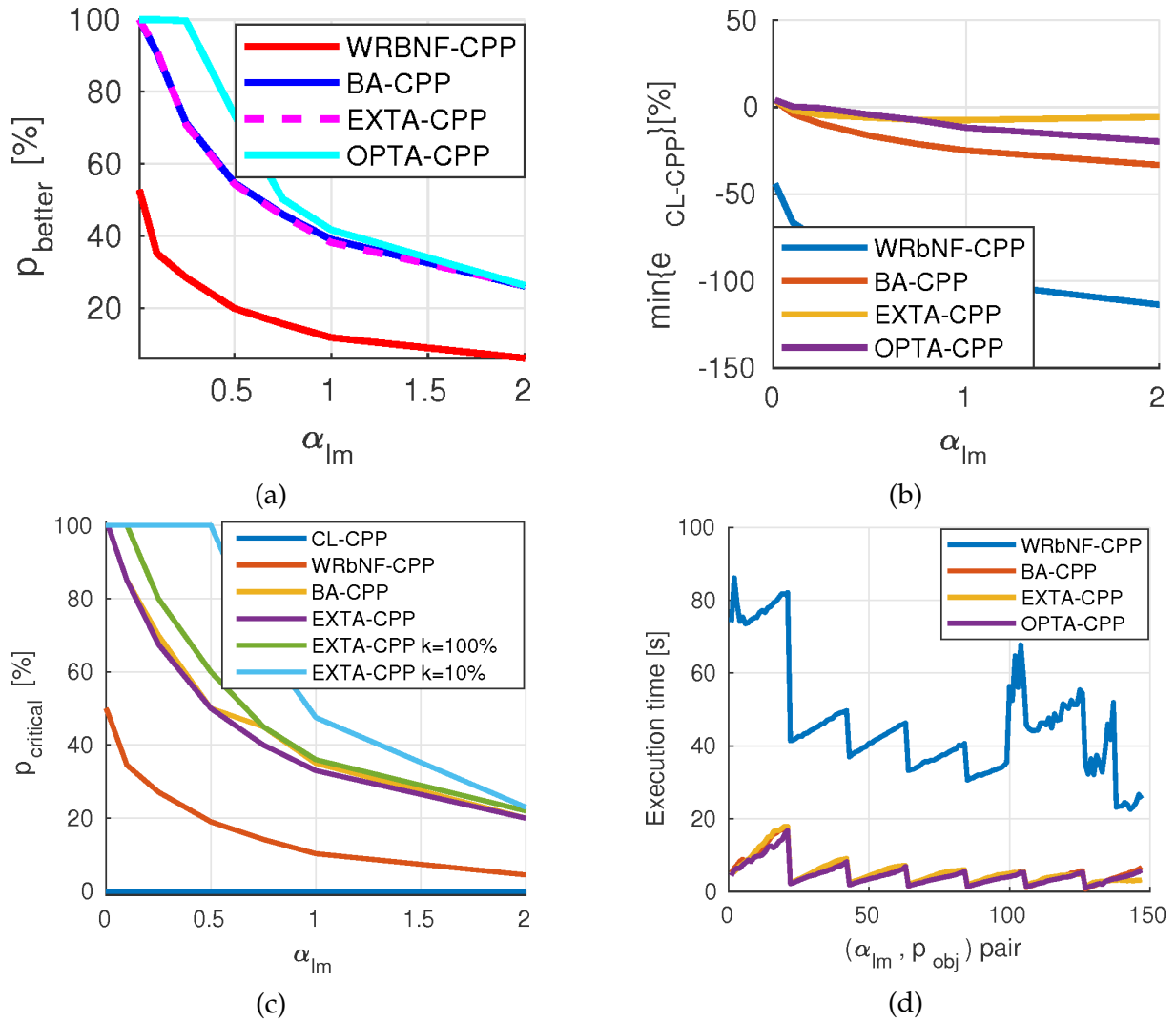
Figure 3.16: (a) Percentage $p_{better}[\%]$ of all simulations ran for each CPP method, for each α_{lm} in which performance of that method was better than the CL-CPP algorithm. (b) Comparison of the worst-case performance of the CPP methods implemented in this chapter. (c) Comparison of the percentages $p_{critical}$ of LM segments containing interesting object(s) needed for the performance of each method presented in this chapter to be the same as the CL-CPP method. (d) Comparison of the execution times of all CPP methods presented in this chapter.

that in most exploration and survey missions, LM segments are significantly longer than wider, and that only a small part of the coverage area is interesting for the mission.

One of the authors' previously published methods named WRbNF-CPP [40], uses path planning to diverge from the initial CPP solution LM pattern in case that it detects interesting objects in it previous LM lane's side-scan sonar image. Other three implemented algorithms, namely BA-CPP published in [39], and EXTA-CPP and OPTA-CPP published in [41], use a decision-making module to replan the coverage path for the rest of the mission based on the sonar data from the previous LM lane.

Upper and lower performance bounds all the coverage algorithms presented in this chapter are estimated and validated through extensive mission parameters variation simulations. Simulation results show significantly shorter coverage paths obtained by all the proposed CPP algorithms compared to the CL-CPP approach, which in a limiting case, results in a twice shorter coverage path. Moreover, OPTA-CPP algorithm shows the best performance in a wide range of mission parameters' values.

This chapter assumed that the coverage path planning module gets interesting objects detections on its input, so it outputs locally replanned coverage path if needed. Next chapter is related to the sonar imagery processing module that processes the collected local sonar data on both sides of the vehicle. This is performed after each passed lane along the full length of the coverage area, so that the output of the sonar imagery processing module is the input of the coverage path planning module.

Chapter 4

Object detection in side-scan sonar imagery

4.1 Introduction

With the rise of high resolution imaging systems comes a need for storing and/or processing enormous amounts of data. Sending these data on-the-go to some centralized ground station for them to be processed is generally speaking not an insurmountable problem for ground and aerial vehicles since WiFi connection is possible. In the underwater environment, however, communication is greatly constrained by the fact that electromagnetic waves are heavily attenuated even at short distances, while at the same time acoustic communications have such a low bandwidth that sending huge amounts of sonar data on-the-fly is still impossible. This is why all the sonar data are stored on an onboard hard or solid state disk when AUVs are used, which allows for only offline processing of sonar imagery, whether by a human operator or by some image processing algorithm. In both cases the data cannot be analyzed online during the mission, so human operator or mission (re)planning algorithm cannot inspect in more detail some interesting objects as soon as they appear in sonar data [77].

In the past few decades AUVs mounted with high resolution side-scan sonars have been increasingly used instead of side-scan sonar mounted towfish. One of their task is to detect interesting objects lying on the seafloor. Automatic computer-aided detection (CAD) and computer-aided classification (CAC) methods are used on sonar imagery for several reasons: to develop consistent and reliable detection/classification algorithms, which would free human operators of often long-lasting and boring task of tagging objects of interest, but also to enable online decision-making for AUVs' mission (re)planning algorithms to react and replan the survey mission to inspect interesting objects in more detail [77]. This way, it would be much more convenient and less
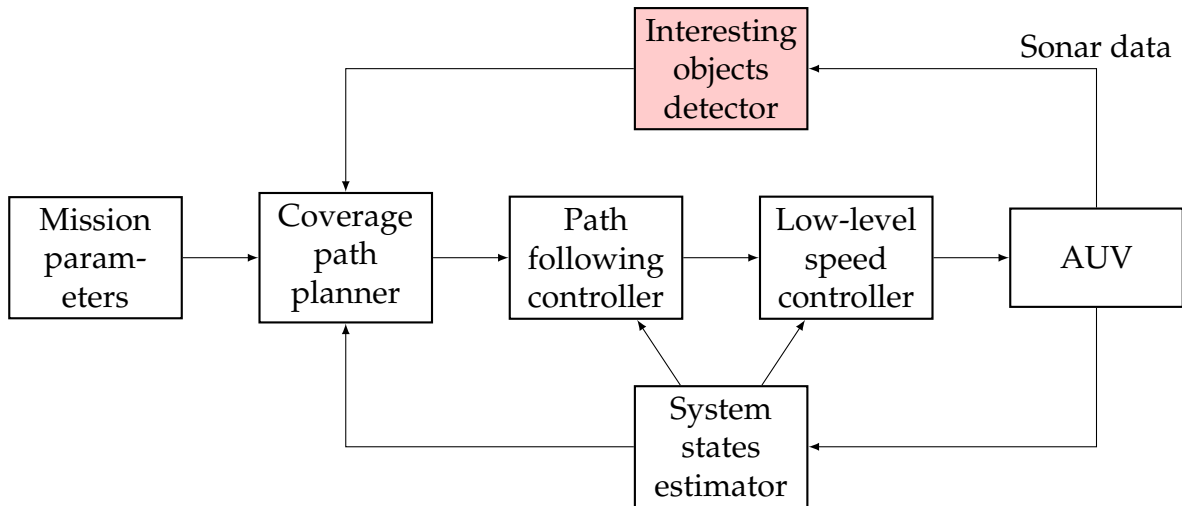
Figure 4.1: Block scheme of the entire AUV system.

costly to deploy an AUV and let it autonomously survey the given area. An AUV (or multiple AUVs) should gather more high-resolution data about parts of the coverage area that it finds interesting for the current exploration mission, and lower resolution general survey data about other parts of the area.

Author proposed several online side-scan sonar data-driven CPP algorithms for monitoring and surveying large-scale (over $1km^2$) seafloor regions by an AUV (in Chapter 3), which would replace all three phases of survey missions when using a towfish. One of the task execution algorithms needed for AUV to autonomously map some area of interest is the module which detects interesting objects in side-scan sonar data, shown in Fig. 4.1 as the "Interesting objects detector" (in red). For a given seafloor area's side-scan sonar image this module should output the coordinates of the interesting objects (if detected) in the area and sends them to the coverage path planning module which then replans the AUV's path in order to inspect those objects in more detail. The goal is to implement all the blocks shown in the scheme in Fig. 4.1 for Lupis AUV. Parts of this chapter were previously published by the author in [78].

The rest of the chapter is organized as follows: a short introduction of what do the terms saliency and anomaly mean, together with an overview of computer vision methods used to model human visual attention is given in Section 4.2. Pipeline developed for using saliency detection methods in the domain of side-scan sonar images is given in Section 4.3. Section 4.4 discusses the performance metrics used to assess how well do the mentioned methods detect salient/anomalous objects in a controlled environment of simulated side-scan sonar image dataset. The results and conclusions of the best method chosen based on the analysis in Section 4.4 are validated on real side-scan sonar image dataset in Section 7.6.1. The method proven to have the best performance is reimplemented to run on the target hardware of AUV Lupis, and its processing time,

crucial for online applications, is analyzed in Section 7.6.2. This chapter is concluded in Section 4.5.

4.2 Overview of saliency methods

4.2.1 Definitions of saliency and anomaly

Making sense of imagery is something that comes naturally to humans, but it remains a challenge to provide a similar capability to computers and robotic systems. Nevertheless, computational image processing has progressed rapidly in the last twenty years enabled by developments in image processing techniques and software and by rapid advances in sensors and computer performance [77]. In order to design algorithms for interesting objects detection in (sonar) images, it is first needed to understand how for example human visual system notices interesting things which pop up from the rest of the visual field.

Human visual system is very selective. This means that we inherently pay more attention to some parts of the scene and neglecting other parts. This is apparent from research studies on change blindness, which show that the scene can be changed even significantly, but not registered by human viewer. The cause of our visual system to be selective lies in the way that our brains process visual information. The optic nerve receives visual information at a rate of approx. 3×10^6 bits/s, the brain processes less than 10^4 bits/s of this information. It is astounding how the brain uses only a small fraction below 1% of the input information to generate a representation of the scene good enough for executing complex tasks in a dynamic environment. But what mechanisms are responsible for such sparse yet very effective representation of the scene [79]?

Two key processes of concentration are discussed in the literature: top-down, and bottom-up. Top-down is voluntary, goal-driven and slow, i.e. usually within the range of 100ms to a few seconds. The top-down attention is believed to be closely linked to cognitive factors such as learning, perception, and reasoning. In comparison, bottom-up focus (also known as visual saliency) is correlated with a scene's qualities which attract our attention to a specific location. These attributes include motion, contrast, orientation, brightness and color. Mechanisms of the bottom-up attention are spontaneous, and quicker than from the top down [79]. Studies show that in search tasks, such as looking for a target object among distractors, both bottom-up and top-down mechanisms work together to guide our attention. While bottom-up attention is based on elementary attributes of a scene, top-down is quite complex and strongly influenced

by task demands [79].

Modeling the visual saliency has created much interest in the research community over the past two decades. It has also paved the way for a number of computer vision technologies, in addition to leading to understanding of human perception. These applications include target detection, image and video compression, image segmentation, context aware image resizing, robot localization, image retrieval, image and video quality assessment, dynamic lighting, advertisement, artistic image rendering and human–robot interaction [79].

To test the efficiency of visual saliency algorithms, the two-dimensional saliency maps are contrasted with those picture regions that attract the attention of observers by showing a set of images to the observers and using an eye tracker to monitor their eye fixations. Therefore, a higher number of fixations was assumed to lead to prominent picture areas. Thus, the reported fixations are correlated in a pairwise manner with the corresponding visual saliency charts. However, experiments have shown that viewers appear to focus more on the center of the image than on the periphery regions when processing pictures. The existence of center bias in fixations makes it difficult to determine the relationship between the fixed regions and the salient regions of the picture [79].

The importance of anomaly detection is due to the fact that data anomalies translate into relevant (and often critical) actionable information in a wide variety of application contexts [80]. It is safe to say that interesting objects in (sonar) images are both salient and anomalous w.r.t. the mostly sandy seafloor around them. Detection of anomalies relates to the problem of finding variations of data that don't correspond to planned behavior. Anomalies and outliers are two words more widely used when identifying anomalies; often interchangeably. Anomaly identification sees widespread use in a broad variety of applications such as credit card fraud prevention, insurance or health care, cyber-security intrusion detection, security critical systems' failure detection and enemy military monitoring [80].

Two wide types of the detection / classification of salient / anomalous artifacts are in use:*supervised algorithms*, involving training data for target items in known locations, and *unsupervised algorithms*. Well-designed, supervised algorithms can be assumed to have superior performance when equipped with sufficient data for specific environments. The main limitation of implementing such techniques is that it is not often possible or easy to obtain appropriate training data sets. The training data must be thorough and obtained under comparable sonar and environmental conditions to those in the data for which object detection is needed [80].

Unsupervised algorithms are constructed in the absence of training data to operate

under a variety of conditions. Thus, they are easier to operationally incorporate without the need for external surveys to collect appropriate training data[77]. Unsupervised saliency map and anomaly detection algorithms applied in the field of computer vision use a number of models, i.e. Bayesian, cognitive, decision theoretic, graphical, information theoretic, pattern classification, spectral analysis, and many other types of models. Interested reader is referred to ([79]) and ([80]) survey papers on unsupervised saliency and anomaly detection methods. Supervised saliency map and anomaly detection algorithms are implemented as artificial neural networks (ANNs) which need exhaustive training data sets to have good performance and generalization under arbitrary circumstances as in [81–83].

4.2.2 Methods for interesting objects detection in side-scan sonar images

Computer-aided detection / classification techniques (CAD / CAC) have been under use of side-scan sonar imaging since the early 1990s. The most efficient strategies are based on a combination of acoustic highlight and shadow correlated with an item lying on the seabed [77]. In the context of detecting interesting objects or targets in side-scan sonar images, several saliency/anomaly detection methods have been developed, both unsupervised and supervised, e.g. [84–96].

4.3 Pipeline for the compared methods

Considering side-scan pings stacked together (in a so-called "waterfall view") as a grayscale image, various anomaly/saliency image detection methods were implemented and tested on simulated and real side-scan sonar images. These methods, except for the anomaly detection in side-scan sonar data [95] were almost exclusively used in camera images, namely Contrast-based saliency method [97], graph-based saliency [98], Itti-Koch saliency [99], and Simpsal saliency [100] as a simplification of Itti-Koch method. It is worth noting that other methods have also been tried out, namely [96], which uses diffusion maps for anomaly detection, but was too slow to be considered for online use on-board an AUV, even though it detected anomalous interesting object quite good. Method for generating image signatures based on highlighting sparse salient objects in an image, [101], was much worse than the other methods when used on side-scan sonar images. Also, translating side-scan sonar grayscale image (or its parts) into its fractal dimension to differ naturally occurring objects and man-made objects based on method [102] was a few order of magnitude slower than the other methods even for

small portions of sonar images.

A general pipeline was developed in order to use any of the five methods on any side-scan sonar image. Pseudocode of this pipeline is given in Algorithm2. All the above-mentioned methods process side-scan sonar image which is preprocessed with nadir removal method (line 2 in Algorithm2), as well as along- and across-track normalization with a moving average filter to compensate for the attenuation of the returned signal towards the edges of the sonar range (line 3 in Algorithm2), and also smoothing of the image to additionally suppress noise (line 4 in Algorithm2). Saliency/anomaly detection methods mentioned above process the image on multiple scales (resolution reduction spaces) in order to detect pixels which locally differ the most w.r.t. their neighboring pixels, given in lines $5-15$ in Algorithm2. After saliency/anomaly map is computed, it is then thresholded w.r.t. its mean value (line 16 in Algorithm2), and then it is segmented into contours (lines $17-19$ in Algorithm2) which are thresholded by their estimated area in 2D space (line $20-22$ in Algorithm2). Finally, detected objects are thresholded by their brightness (line 23 in Algorithm2) w.r.t. the fraction of the maximum brightness in the image, and finally the bounding boxes of those contours are computed for easier representation (line 24 in Algorithm2). The results this pipeline's performance testing are given in the following sections.

Pseudocode of the method for anomaly detection from [95], here named *getAnomalyMap* method, is given in Algorithm3. The histogram difference metric for local variation of a pixel w.r.t. its neighbors through the entire Laplacian-over-Gaussian filtered scaled down originals of the sonar image (lines $4-9$ in Algorithm3), defined in [95] as ℓ_1 norm, is that algorithm's serious bottleneck. It is here approximated as an Nd kernel (lines $10-12$ in Algorithm3), which gives similar results to the original ℓ_1 norm, but in a fraction of the time.

Contrast based saliency method from [97] was originally used for saliency detection in videos. Here it is named *getContrSalMap* and was extended to be used for side-scan sonar images and also on multiple resolution scales just as *getAnomalyMap* method.The only difference in *getContrSalMap* function's code compared to method *getAnomalyMap* is in line 6 (see Algorithm3). Instead of Laplacian over Gaussian filtering applied to each downsampled sonar image, its brightness distance map is computed as described in detail in [97].

4.4 Simulated dataset results

The first step to deciding which method is best used for the problem at hand, a simple side-scan sonar images dataset of 1500 $2MP$ images was generated, mostly as noisy

---

Algorithm 2 Pipeline for anomalous object detection

1: **function** DETECTANOMALIES(img, $params$)
2:      $img \leftarrow img.$**removeNadir**$(params.nadir\_width)$
3:      $img \leftarrow img.$**normalizePings**$(params.mavg\_M)$
4:      $img \leftarrow img.$**smooth**$(params.gauss\_kernel)$
5:      **switch** $params.s\_method$ **do**
6:          **case** $"itti - koch"$
7:              $img \leftarrow img.$**getIttiKochMap**$(params)$
8:          **case** $"gbvs"$
9:              $img \leftarrow img.$**getGBVSMap**$(params)$
10:         **case** $"simpsal"$
11:             $img \leftarrow img.$**getSimpsalMap**$(params)$
12:         **case** $"anomaly"$
13:             $img \leftarrow img.$**getAnomalyMap**$(params)$
14:         **case** $"constrast"$
15:             $img \leftarrow img.$**getContrSalMap**$(params)$
16:      $img \leftarrow img.$**threshold**$(params.saliency\_th)$
17:      $edges \leftarrow img.$**detectEdges**$(params.e\_method)$
18:      $edges \leftarrow edges.$**median**$(params.median\_kernel)$
19:      $contours \leftarrow edges.$**segment**$()$
20:      $areas \leftarrow contours.$**getArea**$()$
21:      $areas_{th} \leftarrow areas.$**threshold**$(params.area_{th})$
22:      $contours \leftarrow contours(areas_{th})$
23:      $contours \leftarrow contours.$**threshold**$(params.brig\_th)$
24:      $bboxes \leftarrow contours.$**getBBoxes**$(contours)$
25:      **return** $bboxes$

Algorithm 3 Multiscale anomaly map

1: **function** GETANOMALYMAP(img,$params$)
2:      $f\_bank \leftarrow zeros(img.size(), params.n_{scales})$
3:      $f\_bank \leftarrow zeros(img.size(), params.n_{scales})$
4:      **for** $i \leftarrow 1$ to $params.n_{scales}$ **do**
5:          $img_{sd} \leftarrow img.$**scaleDown**$(params.scales[i])$
6:          $img_{log} \leftarrow img_{sd}.$**LoG**$(params.log\_kernel)$
7:          $img_{su} \leftarrow img_{sd}.$**scaleUp**$(params.scales[i])$
8:          $img_{su} \leftarrow img_{su}.$**normalize**$()$
9:          $f\_bank[i] \leftarrow img_{su}$
10:      $kernel = ones(3, 3, params.n_{scales})$
11:      $kernel(2, 2, :) = -8$
12:      $anomaly_{map} \leftarrow$ **convolve**$(filter\_bank, kernel)$
13:      $anomaly_{map} \leftarrow anomaly\_map.$**normalize**$()$
14:      **return** $anomaly_{map}$

Table 4.1: Table of interesting object detection methods' parameters for simulated side-scan sonar images dataset.

	Anomaly Detection	Contrast-based Saliency	Itti-Koch Saliency	GBVS Saliency	Simpsal Saliency
mavg_M	10	10	10	10	10
gauss_kernel_w	10	10	10	10	10
gauss_sigma	1.6667	1.6667	1.6667	1.6667	1.6667
saliency_th	7	7	3	3	3
e_method	sobel	sobel	sobel	sobel	sobel
median_w	3	3	3	3	3
area_th	25x25cm	25x25cm	25x25cm	25x25cm	25x25cm
brigh_th	30.00%	30.00%	30.00%	30.00%	30.00%
scales	[1, 3, 5]	[1, 3, 5]			
blur_fraction			0.002	0.002	0.002
feature channels	IR	IR	IO	IOR	IO

grayscale images with a few $(1-5)$ objects and their simulated sonic shadows of various sizes $(1x1, 1.5x1.5, ..., 10x10m^2)$ and position distributions in the sonar image. With the assumed $50m$ across-track range of the side-scan sonar, the resolution of each pixel was taken as $5 \times 5cm$. Performance benchmarking of the above-mentioned five methods for salient/anomalous object detection was performed on a computer with a quad-core $2.8GHz$ CPU with $16GB$ of RAM and $4GB$ Graphics processing unit (GPU). Software used for benchmarking on the simulated dataset was MATLAB with its Image Processing Toolbox and Parallel Computing Toolbox with Compute Unified Device Architecture (CUDA) GPU Processing Support. All five methods used to detect interesting objects in side-scan sonar images were optimized for execution on a GPU in order to perform as fast as possible, having in mind that these methods would later on be used in Gazebo simulator [103] in which Unmanned Underwater Vehicle (UUV) simulator [104] and side-scan sonar simulator [105] environments would be integrated with our sonar image processing, online coverage path planning ([39,40], and [41]), and control modules ([7,8]).

All five methods were applied to the simulated side-scan sonar image dataset, and were compared to the known ground truth of interesting objects pixels indices. Parameters of all these methods and steps in the interesting object detection pipeline described in Section 4.3 are given in Table 4.1.

An example of such simulated side-scan sonar image and five methods' performance in detecting interesting objects standing out in those images is given in Fig. 4.2. It is clear that the anomaly detection method from [95] gives the most accurate and precise detection results compared to other methods. In order to formalize this, a precision-recall analysis was conducted. The so-called F harmonic mean metric was used to aggregate these two metrics, which is defined as:

$$F = 2 \times \frac{precision \times recall}{precision + recall}. \tag{4.1}$$

Fig.4.3a shows the results from all 1500 test conducted. It is clearly visible that at 90% F-measure, the anomaly detection method from [95] is much better than Itti-Koch, Graph-Based Visual Saliency (GBVS), as well as contrast based saliency method. Simpsal is the closest with around 80% F-measure. Fig.4.3b shows execution times of all 1500 runs for each of the chosen 5 methods. It is obvious that Simpsal is much faster than the anomaly detection method which is more precise and accurate in its detections. Taking into account that the target hardware on-board AUV Lupis is roughly $10 - 50$ times slower than a high performance workstation computer used for benchmarking, even then anomaly detection method running $1.5 - 7.5s/MP$ is considered to be fast enough together with its winning precision/recall performance.

4.5   Conclusion

Considering side-scan pings stacked together (in a so-called "waterfall view") as a grayscale image, in this chapter various anomaly/saliency image detection methods were implemented and tested. These methods, except for the anomaly detection in side-scan sonar data [95] and graph-based visual saliency (GBVS) saliency [98] were almost exclusively used in for natural images, namely contrast-based saliency method [97], Itti-Koch saliency [99], and Simpsal saliency [100] as a simplification of Itti-Koch method. The above-mentioned methods were first benchmarked in Matlab on a set of 1500 simulated side-scan sonar images dataset containing a single and also multiple objects of various sizes and at various positions. The anomaly detection method from [95] had the best precision and recall performance on simulated sonar images dataset. Its performance was then validated on a real 500 side-scan sonar images dataset, and it also had satisfactory recall-precision performance. The next chapter describes the ROS simulation framework used for coverage path planning algorithm validation.
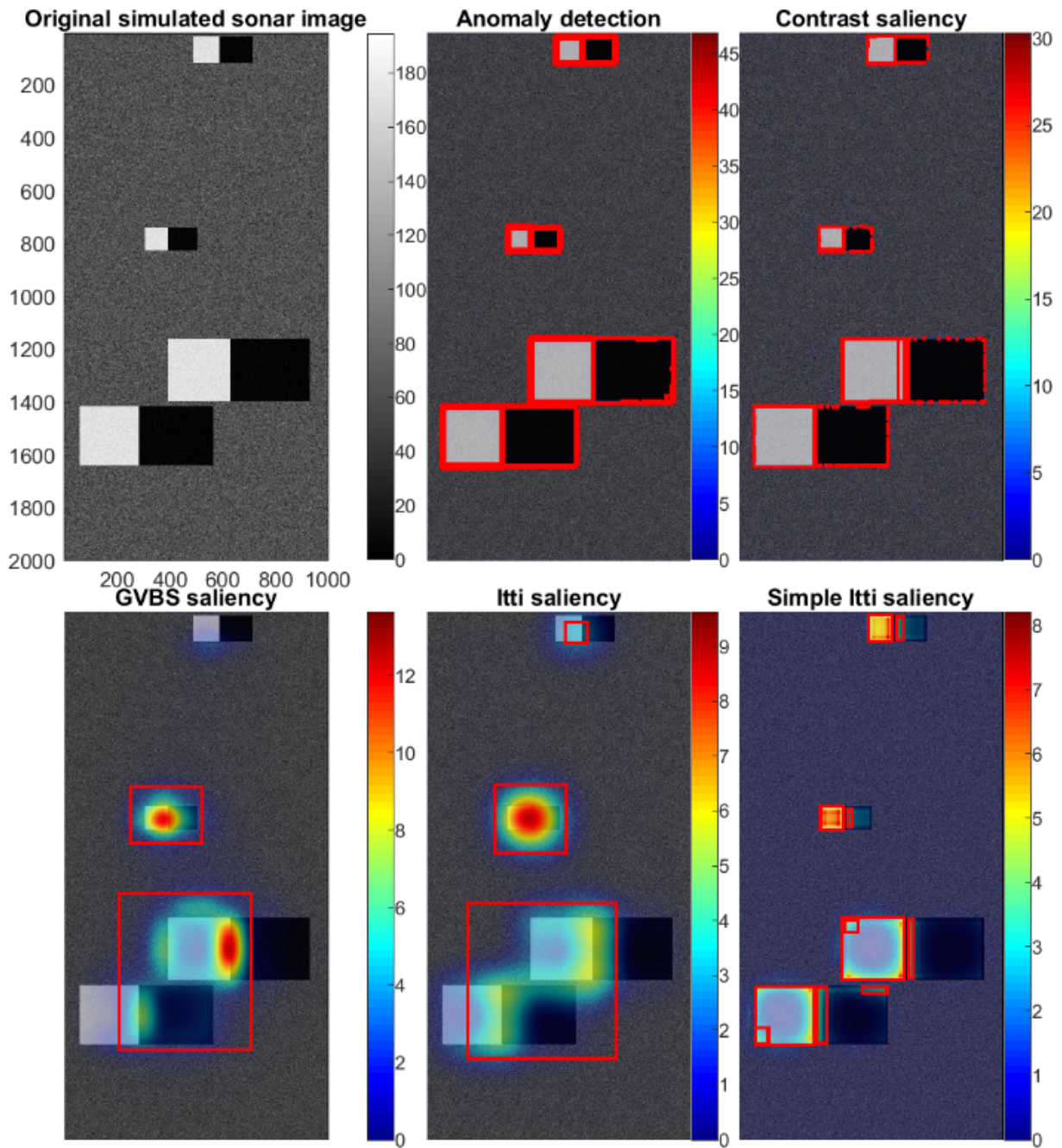
Figure 4.2: An example of interesting objects detection in simulated side-scan sonar data by various image segmentation methods. Colorbar represents the saliency metric normalized by its mean value to visualize which areas are the most interesting. Since anomaly and contrast methods practically detect edges in multiple scales, the anomaly/saliency values in the upper subfigures is covered with detection bounding boxes.
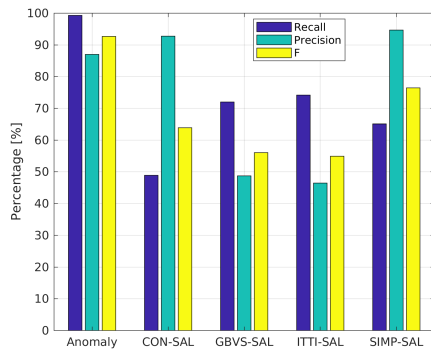
(a)
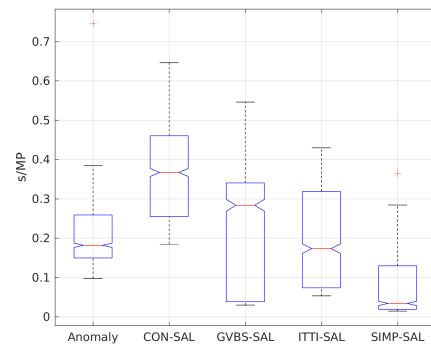
(b)

Figure 4.3: (a) Mean misdetection rates of the chosen anomaly detection methods. (b) Comparison of mean recall, precision, and combined metric usually used in image segmentation/detection methods

Chapter 5

Simulation environment for coverage planning algorithms validation

5.1   Introduction

Sonar data-based algorithms, namely CL-CPP, BA-CPP, and OPTA-CPP, mentioned in Chapter3were implemented in ROS Melodic framework together with side-scan sonar anomaly detection algorithms mentioned in Chapter4. This chapter describes the integration these modules with the open-source UUV simulator that was used as a base for simulation validation environment. UUV simulator is described shortly in Section5.2. Section5.3presents the implementation of the above-mentioned CPP algorithms as well as the coverage mission controller module. Examples of coverage missions using CL-CPP and OPTA-CPP algorithms together with performance comparison is given in Section5.4. Section5.5concludes this chapter.

5.2   UUV simulator

UUV simulator [106] is a ROS package that contains ROS nodes and Gazebo plugins for underwater robotics simulations of AUVs and ROVs. It was developed in the scope of EU ECSEL SWARMs project. It is currently released for ROS Kinetic, Lunar, and Melodic. The simulator offers a few Gazebo worlds to be used in different use-cases, namely

- Empty underwater world - generates a world model with 3D current velocity topics to simulate the effects of water current disturbances on vehicle control,
- Herkules shipwreck - offers a 3D reconstruction of M/S Herkules including blue fog to emulate decreased visibility,
- Lake - with uneven lake floor topology,

- Coast of Mangalia, Romania - with an approximated seafloor topology to that in the vicinity of Mangalia,
- Munkholmen, Norway - recreation of a site close to the final SWARMs demonstration scenario,
- Ocean waves world - developed, as the name suggests, to animate waves during simulation, and
- Subsea BOP panel - another scenario that was used for SWARMs project demonstration.

Moreover, it is possible to generate custom-made world models for any kind of simulation. Manual control is enabled for RexROV and RexROV2 unmanned underwater vehicles.

Apart from that, tuned low-level PID and sliding mode controllers enable the use of various ROS services such as

- AddWaypoint - adds a Waypoint at the end of the existing WP list,
- ClearWaypoints - removes all WPs from the current WP list,
- GetWaypoints - returns WP list,
- GoTo - command the vehicle to go to the defined absolute position in ENU/NED frame, with defined maximum velocity, start time, and path interpolator
- GoToIncremental - command the vehicle to go to the defined relative position w.r.t. the vehicle's current position in ENU/NED frame, with defined maximum velocity, start time, and path interpolator
- Hold - starts station keeping mode; for fully actuated ROVs this means dynamical positioning, while for underactuated AUVs it keeps them inside a predefined circle to prevent to big of a drift,
- InitCircularTrajectory - initiates a circular trajectory with defined center in ENU/NED frame, radius, and start time,
- InitHelicalTrajectory - initiates a helical trajectory with defined center in ENU/NED frame, radius, number of turns, depth increments, and start time,
- InitWaypointSet - starts a trajectory based on Waypoint list that is passed to the service at call, together with the desired start time, maximum velocity, and path interpolator
- InitWaypointsFromFile - the same as InitWaypointSet, but the WP list is read from a predefined .yaml parameter file.

This means that any path planning and perception algorithms can easily be built on top of this existing framework. **InitWaypointSet** ROS service was mostly used by the coverage mission controller to command the vehicle to follow the (re)planned path. **Hold** service was used to keep the vehicle in place before starting the replanned path

to preserve its stability.

Furthermore, the reference to the controllers are computed by using path interpolators and then finite differentiation for velocity and acceleration set-points, in case they are desired. This is done in order to achieve smooth paths, except in the case of linear interpolator. Path interpolators that are available, and are passed as parameters in ROS service requests, are:

- 'linear',
- 'cubic' - uses a cubic Bézier curve,
- 'lipb' - Linear interpolator with polynomial blends - linear segments between each two WPs with fifth order Bézier curves interpolating the corners,
- 'dubins' - computes the shortest curve between two points.

UUV simulator also provides disturbance generators that can be used to test control algorithm performance in the presence of thruster failure/reduced efficiency, or sea currents. Disturbance manager node enables these disturbances to be configured and started at desired times.

Gazebo and RViz views after running an exemplary PID controller scenario are given in Figs.5.1and5.2. RexROV2 vehicle is fully actuated with 3 thrusters for heave control in vertical plane, two thrusters at $45°$ angle in horizontal plane for yaw control, and one lateral thruster for sway control. This means that it's allocation matrix has full rank and is directly invertible. It has GPS, IMU, DVL, particle concentration sensor, magnetometer, pressure sensor, and 3 cameras (front, left, and right) looking forward with a downward tilt. This sensor suite is extensible to any Gazebo sensor plugins that are available. Furthermore, these senors can be used for sensor fusion in a state estimation node, e.g. of ROS *robot_ localization* package, which is out of the scope of this thesis.

Node graph of this scenario is shown in Fig.5.3. Node *gazebo* is responsible for simulation and visualization of world and vehicle physics it send ground truth position of the vehicle in local inertial frame to the */pose_ gt* topic. Node *rov_ pid _ controller* is responsible for controlling the vehicle in both manual and automatic control mode. When in manual mode, it transforms the values from joystick axes to changes in position reference. In automatic or trajectory tracking mode, it transforms the position error based on the vehicle position from */pose_ gt* topic (that can easily be switched to an output topic of some state estimator node) and the reference trajectory computed by trajectory generator upon calling one of the above-mentioned ROS services to a vector of forces and torques that the thrusters need to achieve. Node *thruster_ allocator* performs thruster allocation based on the thruster allocation matrix (TAM), the input vector of desired forces and torques, as well as thruster limits. These values are then

Figure 5.1: Gazebo view of RexROV2 vehicle in world model with loaded seafloor and waves.

Figure 5.2: RViz view of RexROV2 vehicle.

sent to */thruster/No./input* topics that the *gazebo* node subscribes to in order to spin the thrusters at the given rate. Node *trajectory_ marker_ publisher* subscribes to topics related to the waypoints and the interpolated reference trajectory and transcribes them to RViz-readable format for visualization in RViz as well.

5.3 ROS implementation

5.3.1 Coverage planner

Nonadaptive CL-CPP algorithm, as well as sonar data-based coverage (re)planning algorithms BA-CPP and OPTA-CPP, described in detail in Chapter 3 are implemented in Python, and present by the class diagram in Fig. 5.4. As the lawnmower pattern generating algorithm is at the core of all three above-mentioned algorithms, it was implemented as the bsae class named *LawnmowerCoveragePathGenerator*. It takes various ROS parameters and initializes its attributes to those values. e.g. coverage area length, width, and heading, starting waypoint, length of coverage area, offset length, as well as sonar range (swath in the horizontal plane) and coefficient of lane width in terms of sonar across-track swath. It initially generates all waypoints (WPs) in NED frame (with constant depth as the input parameter) assuming that the vehicle moves from the frame origin forward along the coverage area, and turns to the right across the cover-

Figure 5.3: RViz view of RexROV2 vehicle.

age area so that the axes of the coverage area coincide with the axes of the local inertial frame. After generating a waypoint set in this manner, axes flipping (in case that the vehicle should cover the area turning always to the left, or if the simulator like Gazebo uses ENU frame), rotation by the specified coverage area heading angle, as well as translation to the desired starting WP in NED frame are performed, respectively.

BA-CPP algorithm implemented as BasicAccordionCoveragePathPlanner class inherits LawnmowerCoveragePathGenerator witout any additional attributes since it initially assumes that the coverage area does not have any interesting objects and thus start with the LM lanes twice wider than the sonar swath. It thus reimplements method *plan _ coverage* so that if any anomalies are detected on the left and/or right side of the vehicle it replans the coverage path locally (methods _ *replan_ coverage_ path_ locally_ caseX* in Fig.5.4) to adapt to the newly available information, but generates the twice wider LM coverage pattern for the remainder of the coverage area. As in LawnmowerCoveragePathGenerator, all the planning is done in NED frame from its origin with zero heading and assuming rightward turns. For replanning purposes, the current and previous coverage WPs, as well as coordinates of the aggregated anomaly detections are transformed back to this frame, and after replanning the new local and remaining global coverage path are transformed with a flip-rotate-translate sequence.

Since OPTA-CPP algorithm is based on BA-CPP algorithm with slight changes when the vehicle must perform a loopback into the part of the coverage area that it already passed, class OptimizedAccordionCoveragePathPlanner inherits BasicAccordionCoveragePathPlanner with only a few method redefinitons for local replanning

for cases 1, 4, 5, and 6, as shown in Fig.5.4.

5.3.2   Mission controller

Mission controller is also implemented as a Python class MissionController run inside a ROS node. Its flowchart is shown in Fig.5.5. It initializes with loading ROS parameters such as coverage planner name (CL-CPP, BA-CPP, or OPTA-CPP), and parameters related to the trajectory generation such as maximum forward speed, radius of acceptance, and path interpolator. Depending on the picked coverage planner type, an object of the corresponding planner class is instantiated for use during the mission.

Initial LM coverage path is generated by the planner and InitWaypointSet ROS service is called so that the vehicle generates an interpolated trajectory and starts following the first WP at the end of the first full LM line. When the starting offset WP is reached, the controller notifies the Sonar data collector module to start stacking sonar swaths into a waterfall image. Offset WPs are added, thus elongating the total coverage lane length so that fan out effect in sonar data due to the vehicle turning at the beginning of each lane is removed. Reaching the reference path's WP is most of the time unattainable, since the interpolated smooth trajectory doesn't perfectly coincide with the piece wise linear path. Thus, the vehicle is said to have reached the current WP when it comes closer than $1m$ of the victory radius around the trajectory point closest to the current WP.

As the ending offset WP for the current full length LM lane is reached, a stopping signal is sent to the Sonar collector module, which then sends the sonar imagery to Sonar data processor that send back number and spatial limits of the detected anomalies, if any. As the vehicle reaches the end of the current lane, it updates lane coverage for the lanes on its left and right side. Only anomalies detected in lanes that haven't been covered more than once are taken into account, and the others are rejected. This prevents the vehicle from performing infinite looping along the same lane. The coverage planner then replans the coverage path locally in an adaptive way, and the remainder of global coverage path is again planned nonadaptively.

After that new trajectory is generated and the vehicle follows the replanned path that always starts with a WP across-track the coverage area. When the across-track WP is reached, depending on the replanning case, the vehicle can continue tracking the next WP that is placed at the end of the following full length LM line, or it can (only in case of OPTA-CPP algorithm) perform a loopback along a portion of a line parallel to the current LM line. After this WP is reached, coverage numbers for both left and right lane are increased by one. These state transitions happen cyclically depending on the type of planner used and the distribution of lanes with interesting

OptimizedAccordionCoveragePathPlanner

+ _replan_coverage_path_locally_case1()

+ _replan_coverage_path_locally_case4()

+ _replan_coverage_path_locally_case5()

+ _replan_coverage_path_locally_case6

+ _loopback_left()

BasicAccordionCoveragePathPlanner

+ __init__()

+ plan_coverage()

+ _replan_coverage_path_locally()

+ _replan_coverage_path_locally_case1()

+ _replan_coverage_path_locally_case2()

+ _replan_coverage_path_locally_case3()

+ _replan_coverage_path_locally_case4()

+ _replan_coverage_path_locally_case5()

+ _replan_coverage_path_locally_case6()

+ _convert_to_rightward_x_frame()

+ _get_direction()

+ _transform_limits()

+ _test_coverage_planner()

LawnmowerCoveragePathGenerator

+ namespace: String

+ base_link: String

+ _coverage_start_wp: List

+ _desired_depth: Double

+ _lines_offset_elongation: Double

+ _lines_length: Double

+ _coverage_area_width: Double

+ _coverage_area_heading: Double

+ _coverage_acrosstrack_direction: String

+ _enu_frame_used: Boolean

+ _sss_range: Double

+ _k_sss_range: Double

+ _debug_plot: Boolean

+ _debug_coverage_planner: Boolean

+ __init__()

+ _get_ros_params()

+ _initialize_class_attributes()

+ plan_coverage()

+ _get_global_lanwmower_coverage_path()

+ _get_waypoint_forward()

+ _get_waypoint_right()

+ _get_waypoint_backward()

+ _get_waypoint_left()

+ _is_valid_waypoint()

+ _rotate_waypoint_list()

+ _flip_waypoint_list()

+ _translate_waypoint_list()

+ _transform_waypoints_frame()

+ _plot_coverage_path()

+ _test_coverage_planner()

Figure 5.4: UML class inheritance graph for CL-CPP, BA-CPP, and OPTA-CPP coverage mission planners

objects. After each path replanning the vehicle is put into station keeping mode for $2s$ to ensure its stability before continuing movement along the new coverage path. Moreover, the vehicle automatically goes into station keeping mode when it reaches the last WP in its trajectory. Thus, the mission is finished if the vehicle reached the last full along/across/loopback WP, and it is station keeping for more than e.g. $5s$.

5.3.3 Integration of subsystems

Mission controller is integrated into the UUV simulator as depicted in Fig.5.6. It is a part of a *Supervisory subsystem* together with *Coverage planner* that is contained in the Mission controller. As mentioned previously, *Monitoring subsystem* is RViz node that, depending on the usecase, visualizes the vehicle, its path, trajectory, sensor readings, and velocity vector. It gets data about the set coverage path from the Mission controller block, trajectory from the *Control subsystem*, and actuator/sensor data from *Gazebo* physics simulator. When Mission controller gets WPs from the Coverage planner, it calls ROS services (inside Control subsystem block), which in turn call *Trajectory generator* block. Generated reference trajectory is passed on to the *Pose controller* block, after which *Control allocator* remaps reference forces and torques into reference RPMs for the thrusters and sends those values to *Thrusters* block inside *Physics subsystem*.

Feedback loop sends sensor measurements from Gazebo to *Sensory subsystem* in ROS. *Navigation* block fuses all sensor data to estimate vehicle pose. Here a ground truth pose of the vehicle was used in the feedback loop. A case study on using navigation filter is given in Section7.3. For faster testing purposes and due to software versioning problems of a SSS simulator in UUV simulator, anomaly detections from block *Sonar processing* are here implemented as randomly appearing, while a case studies on this block and its HIL testing are given in Chapter4and Section7.6.

5.4 Results

An example of CL-CPP and OPTA-CPP algorithms performance validation in realistic UUV simulator environment with simulated RexROV2 dynamics is presented in this section. An area of $200 \times 200m$ is chosen with sonar across-track swath of $r_{sss} = 10m$ and 8 LM lanes of 20 lanes in total containing interesting objects. Maximum vehicle velocity was set to $1m/s$ as instructed in the UUV simulator.

For CL-CPP, Figs.5.7a-5.7cshow timeplots of vehicle's control states. Since no replanning is performed, the vehicle is in automatic and trajectory tracking mode for the whole duration of the mission, while in station keeping mode just before and after com-
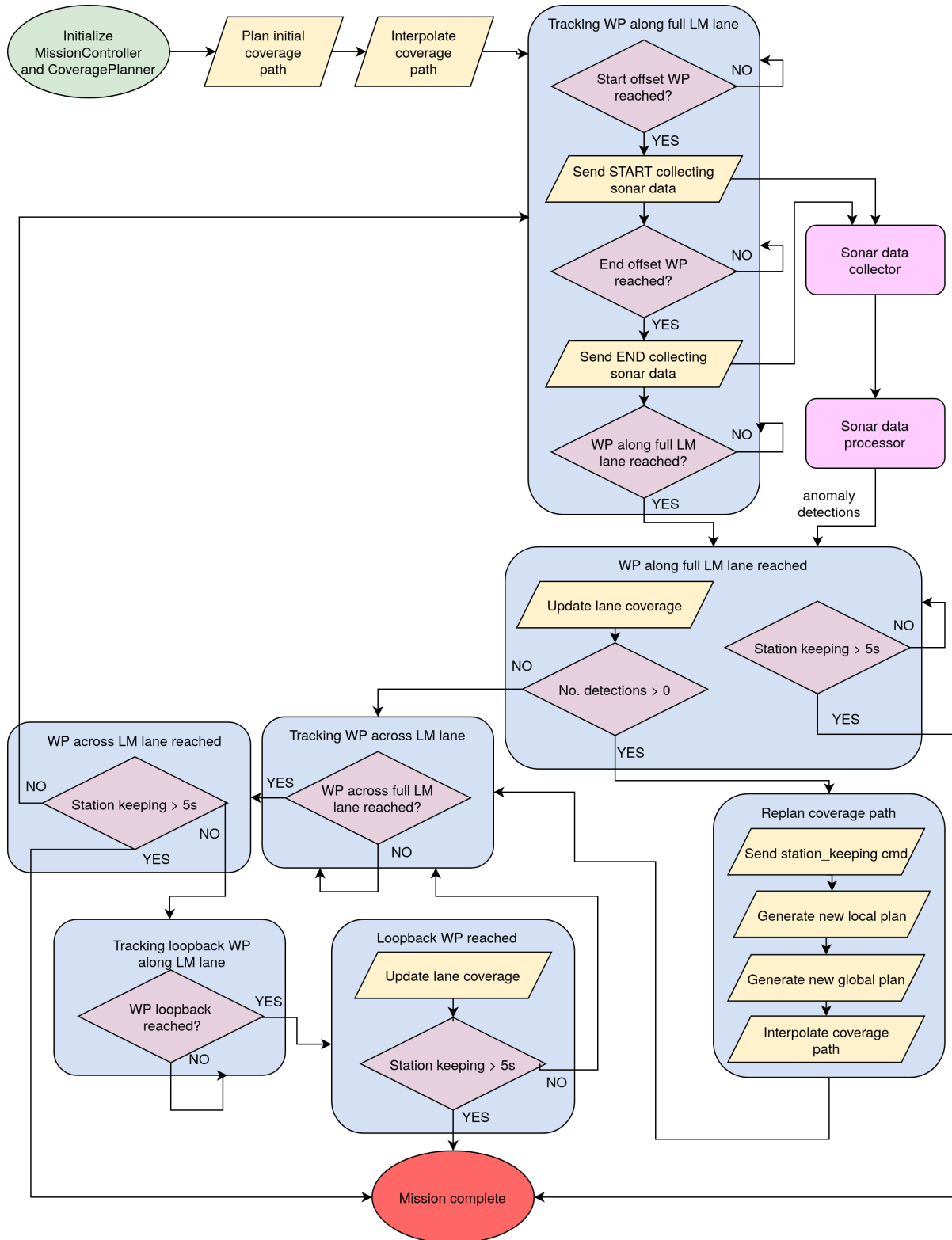
Figure 5.5: Coverage mission controller flowchart. (green) Mission start. (yellow) various routines called during the mission. (blue) Mission controller states. (purple) Decision points. (pink) Sonar data-related outer modules. (red) Mission end.
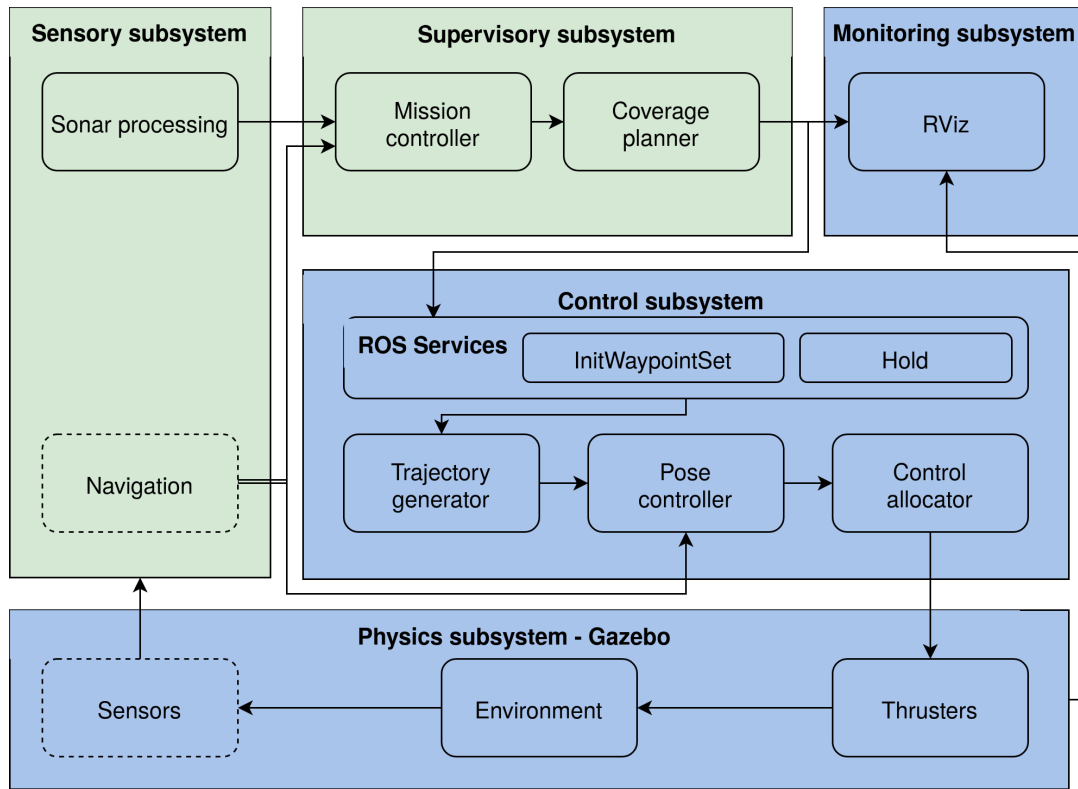
Figure 5.6: Block scheme of software integration of various subsystems. (blue) UUV simulator ROS nodes. (green) ROS nodes implemented in the scope of this thesis

pleting the mission. Time to target estimate, shown in Fig.5.7dshows a monotonous linear decrease of estimated mission duration because no replanning is done. Pose plot of vehicle in North-East plane is shown in Fig.5.8where 'lipb' interpolation of the path between the set WPs can be noticed as the vehicle follows it. It can be seen that the coverage algorithm generates offset WPs to ensure that the vehicle has enough space to make turns and proceed to the next LM line. Vehicle position is shown in a gradient coloring from blue to red hues to show the time component for easier interpretation.

In case of OPTA-CPP, an example of vehicle moving along a line and detecting something interesting on its right side, thus replanning the coverage path visualized in RViz is shown in Fig.5.10. For that same example, Figs.5.11a-5.11cshow timeplots of vehicle's control states. Station keeping mode for a short time during the mission represents times when the vehicle replanned its coverage path and paused to continue moving along the new trajectory in a stable manner. This is shown in Fig.5.11cwhere the vehicle performs replanning for 6 characteristic cases depending on if it moves "forward" or "backward", and it detected interesting object(s) to its left/right/both side(s). Estimated time to mission completion is shown in Fig.5.11dwhere the estimated increases a little with every coverage replanning. In the end, however, the adaptive mission lasts around $1200s$ or 27% shorter than the nonadaptive one, even with 40%
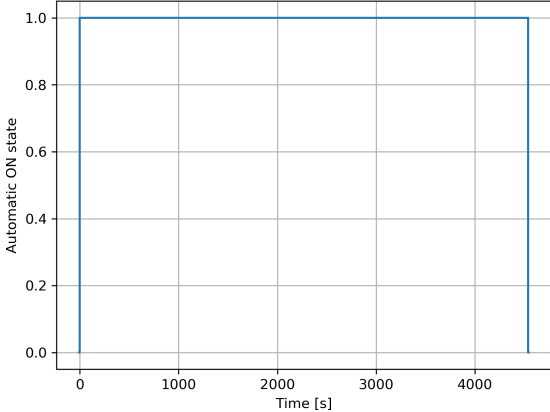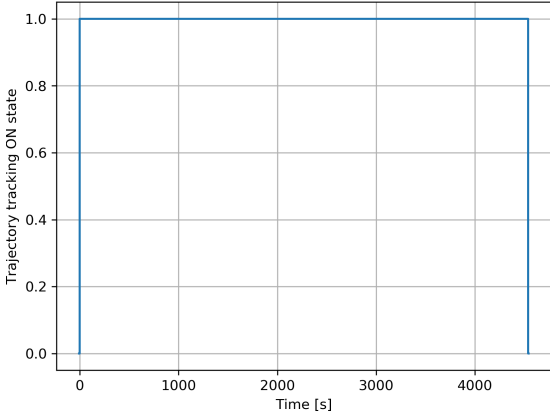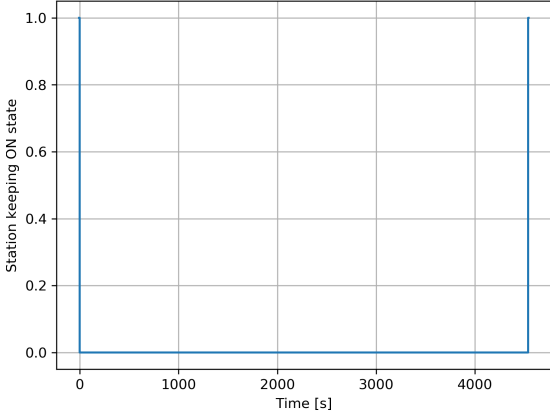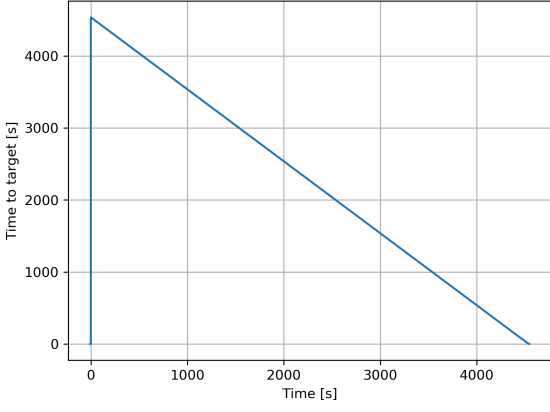
Figure 5.7: Vehicle control states for OPTA-CPP mission. (a) Automatic control mode. (b) Trajectory tracking mode. (c) Station keeping mode. (d) Time to target.
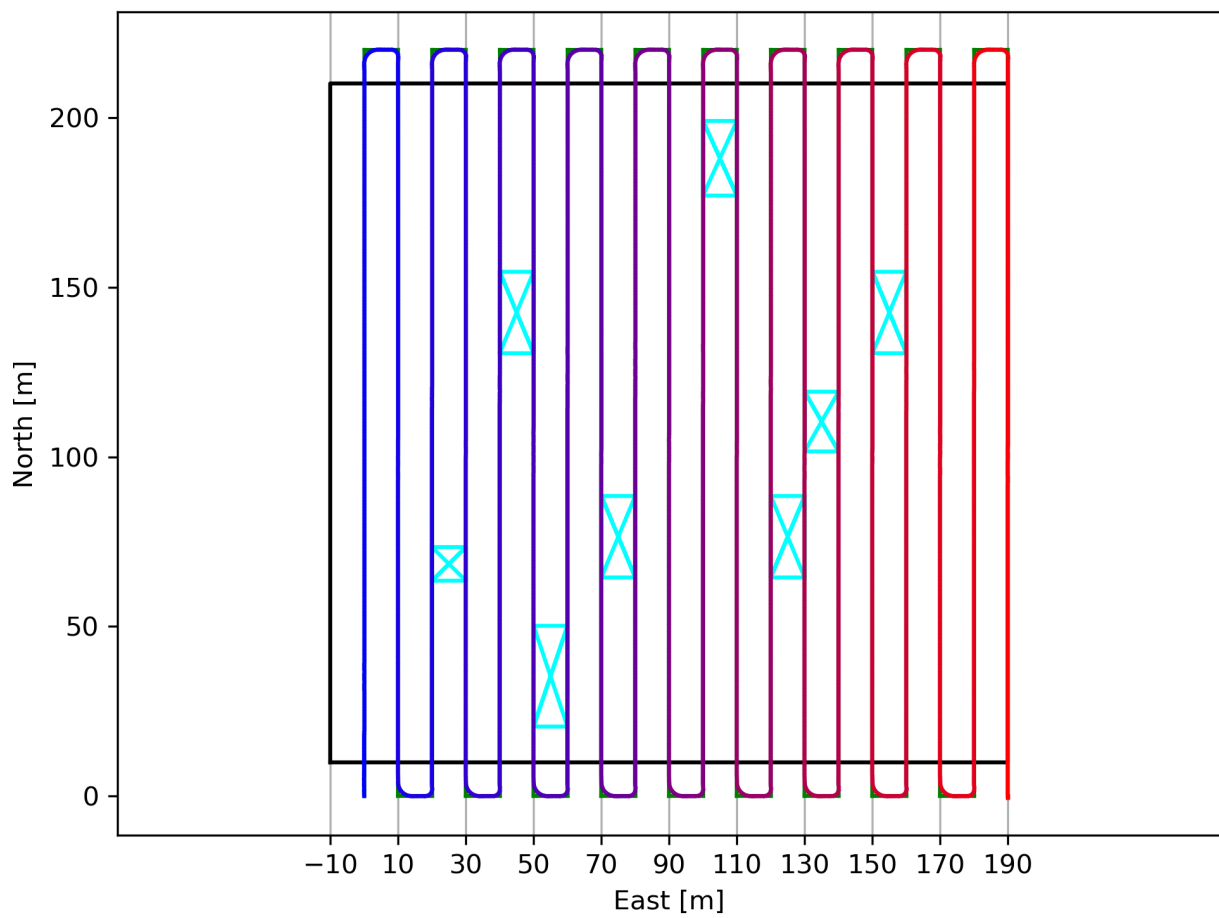
Figure 5.8: Pose plot in NE plane for CL-CPP. (black) coverage area. (cyan) randomly generated interesting objects. (green) coverage path. (blue-red) time color-coded vehicle pose.
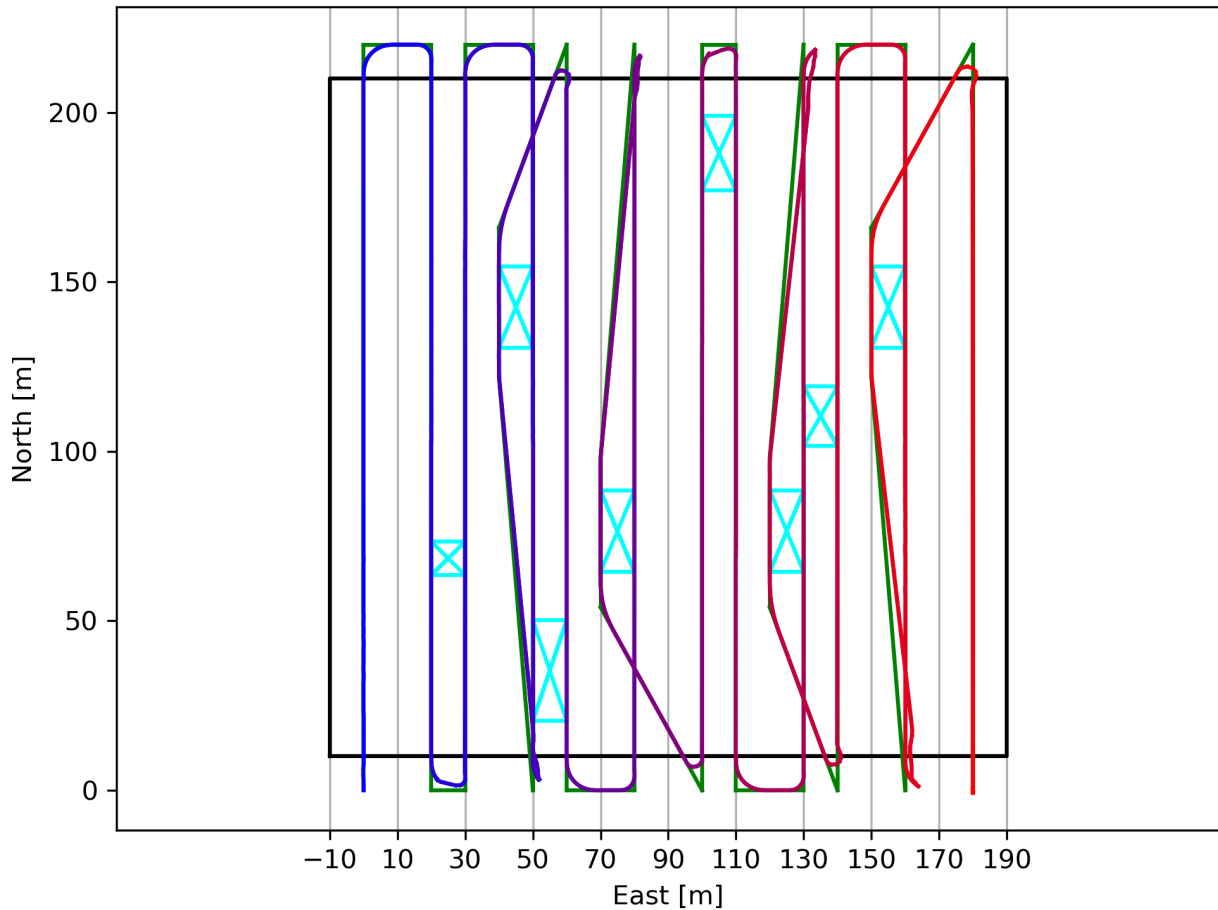
Figure 5.9: Pose plot in NE plane for OPTA-CPP. (black) coverage area. (cyan) randomly generated interesting objects. (green) coverage path. (blue-red) time color-coded vehicle pose.

of LM lanes containing interesting objects. Pose plot of vehicle in North-East plane is shown in Fig.5.9.

CL-CPP and OPTA-CPP performance can be compared either in terms of the total distance traversed for the complete coverage, or the rate of coverage. Fig.5.12shows that the complete coverage path of CL-CPP algorithm is $4543m$ while for OPTA-CPP this is $3568m$, which is performance improvement of 21.46%. Fig.5.13compares the rate of convergence to complete coverage. Coverage percentage was formulated as the percentage of lanes covered at least once. For CL-CPP it is linear since no online replanning takes place. For OPTA-CPP this curve starts off with a twice faster rate but then slows down only a bit. It is constant in cases when the replanned coverage path included a loopback to cover the previous lane once more. CL-CPP finishes after $4320s$ while OPTA-CPP finishes after $3080s$, which is 28.70% performance improvement timewise.
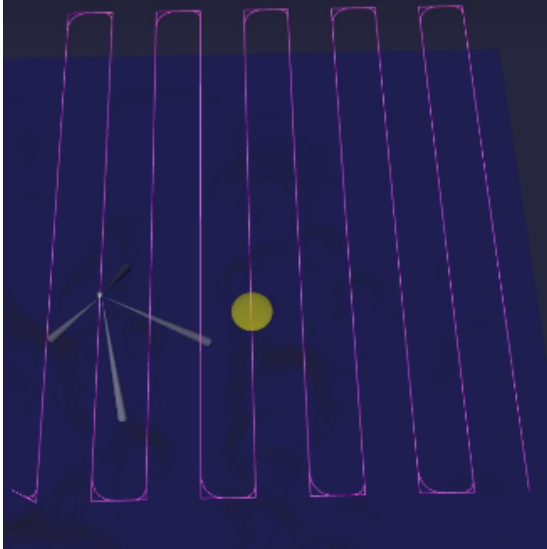
(a)
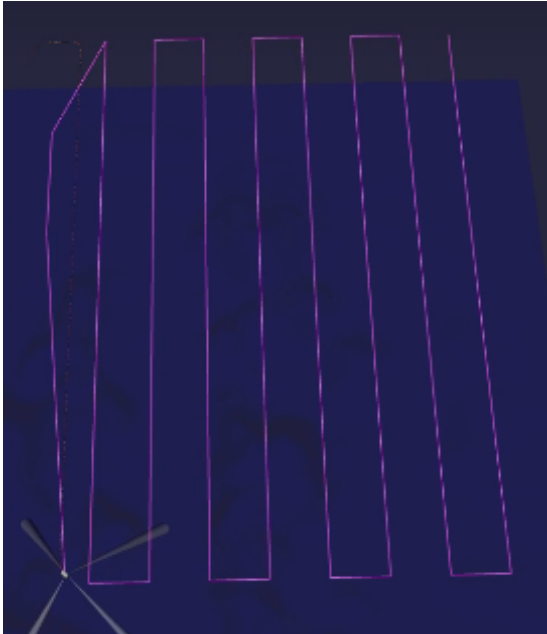
(b)

Figure 5.10: Visualization of mission replanning in RViz. (a) Vehicle moves along the line to the right of which there is something interesting. (b) Vehicle starting to move along the replanned coverage path.
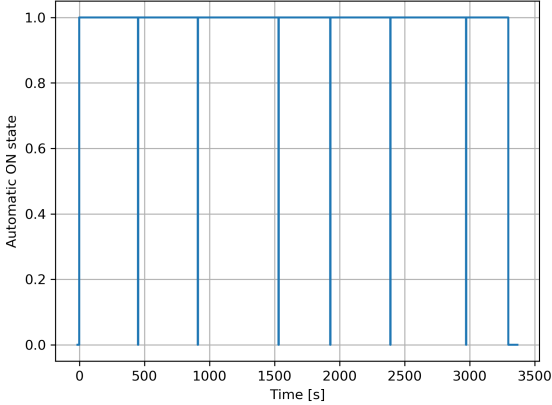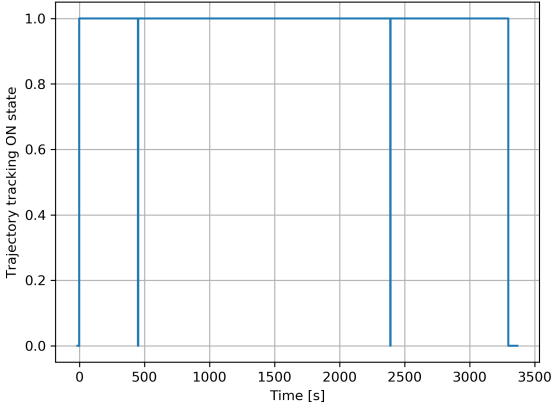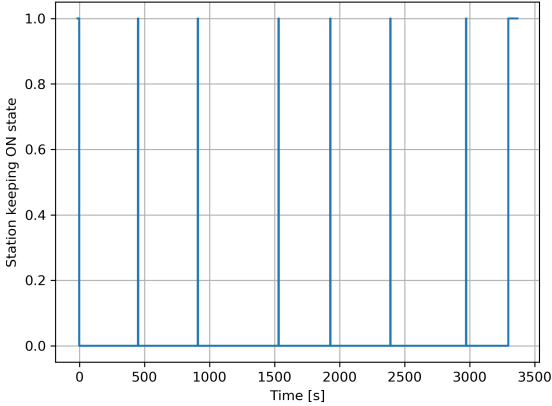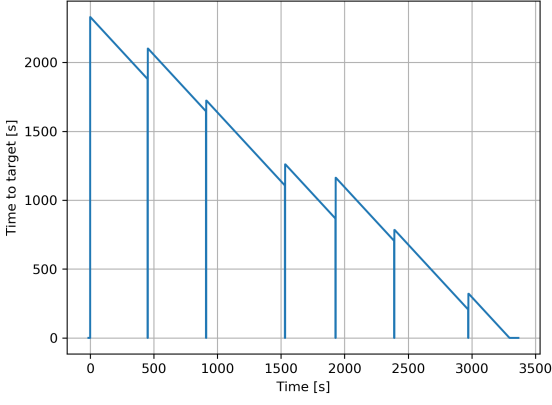
Figure 5.11: Vehicle control states for OPTA-CPP mission. (a) Automatic control mode. (b) Trajectory tracking mode. (c) Station keeping mode. (d) Time to target.
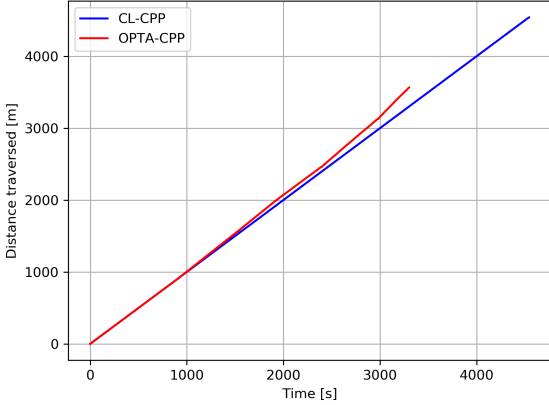
Figure 5.12: Comparison of distances traversed for CL-CPP and OPTA-CPP algorithms to reach complete coverage.
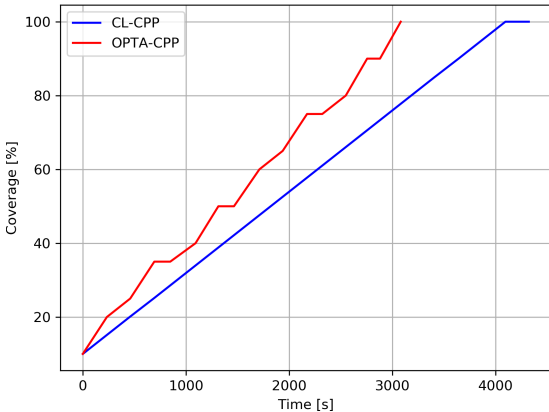
Figure 5.13: Coverage over time CL-CPP and OPTA-CPP algorithms.

5.5   Conclusion

This chapter introduced the UUV simulator as a framework into which the developed coverage path planners and mission controller were integrated. Mission controller is described and represented by a flowchart, and all the subsystems are presented as a block scheme. Results shown here are consistent with the results presented in Chapter 3. The next chapter describes the autonomous marine vehicles (AUV, ASVs, and a hybrid ROV/AUV) that were used in experimental validation trials.

Chapter 6

Systems

6.1 Introduction

Autonomous marine vehicle play an important role in today's seafloor mapping and exploration missions and this trend is only going to increase over the coming years. This chapter brings technical descriptions of the various vehicles used in real-world case studies presented in Chapter7. These vehicles are: LAUV Lupis, described in Section6.2acquired by our research laboratory from OceanScan company; ASV Pla-DyBath, fully developed by LABUST in the scope of BLUEMED project, as described in Section6.3; ASV Korkyra, developed also in-house in the scope of HEKTOR project, as a significantly larger version of ASV PlaDyBath, as described in Section6.4; and a hybrid ROV/AUV Europe, developed by our colleagues at CNR, Genova, Italy, as described in Section6.5.

6.2 LAUV Lupis

Lightweight Autonomous Underwater Vehicle (Lightweight Autonomous Underwater Vehicle (LAUV)) Lupis has been purchased from OceanScan – Marine Systems and Technology, Lda. The LAUV system was originally developed by the Underwater Systems and Technology Laboratory (LSTS) from the Porto University and has been further developed in cooperation with OceanScan – Marine Systems & Technology, Lda. The complete LAUV system includes all the equipment required to communicate with the vehicle, the command and control software, external aids for navigation, and a set of optional devices to facilitate operation. LAUV Lupis is a lightweight, modular platform prepared to integrate a set of different sensors and sonars. The vehicle is targeted at cost-effective oceanographic, environmental and inspection surveys to fulfill a wide range of applications.

Figure 6.1: LAUV Lupis.

On the software side, LSTS developed Inter-Module Communication Protocol (IMC), which defines a common control message used by all vehicles and computers in a networked environment. Unified Navigation Environment (DUNE) is a unified navigation environment, i.e. an embedded software with modules for control, navigation, simulation, networking, sensing and actuation. Neptus is the command and control framework, used to interface and control the LAUV vehicle. It includes two distinct tools:

- The Operator Console supports the planning, execution, and (simplified) review phases.
- The Mission Review and Analysis (MRA) is the interface used to visualize and analyze data recorded by the LAUV vehicle.

6.2.1 Klein UUV3500 Side-Scan Sonar

Klein's UUV-3500 Sonar is a high resolution, dual frequency (400 and $775kHz$) side-scan sonar which uses wideband signal processing techniques to produce superior resolution at long range. It has maximum slant range of $100m$ at $775kHz$ and $200m$ at $400kHz$, and it offers across-track resolutions of $1.2 - 19.2cm$ depending on the transmitted pulse length. Low power consumption is achieved through the use of long, frequency modulated Chirp transmissions while integrated hardware signal processing produces high fidelity acoustic images. The sonar electronics are packaged as an integrated part of an AUV electronics assembly. The sonar transducers mount on the exterior of the AUV, and are configured through-hull penetrators.

6.3 ASV PlaDyBath

An ASV equipped with a Norbit WBMS 400/700KHz multibeam echosounder/sonar (MBES) and accompanying Applanix navigation system together with a high-precision

<div align="center">(a)</div>

<div align="center">(b)</div>
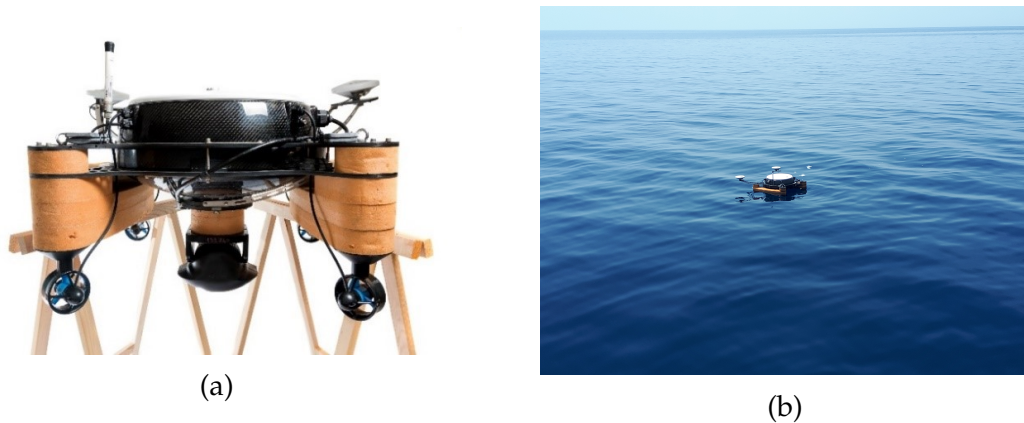
Figure 6.2: (a) Autonomous surface vehicle (ASV) PlaDyBath with sonar mounted below, Trimble GPS antennae in the back and a WiFi antenna on the left. (b) ASV PlaDyBath during a monitoring mission.

Trimble GPS antennae was used to collect multibeam sonar data. This is one of the many application dependent versions of the so-called Dynamic Positioning Platforms (PlaDyPos or H2Omni-X), see Fig.6.2a. The surface vehicle has been developed by LABUST and is used for a variety of applications, from support to underwater archaeology [107], as a dive monitoring platform that allows divers to navigate and monitor from the surface [108], as communication router between underwater and aerial vehicles [109], used in ASV swarms for long-term monitoring of the underwater environment [110], for mapping (obtaining photomosaic and bathymetry) of shallow water areas [111] and for mine countermeasures [112].

The ASV is fully actuated with four electric thrusters that make up the X configuration. This configuration allows horizontal movement in any orientation. The ASV has a diagonal length of $1m$, is $0.35m$ high and weighs about $20-30kg$ depending on the payload configuration in the experiments. The maximum speed under ideal conditions is 1 m/s. Such a configuration of the vessel is very well suited for research purposes due to its simple deployment procedure, robustness under real environmental conditions and low energy consumption, [108,113].

6.3.1 Multibeam sonar Norbit iWBMSe

The Norbit iWBMSe multibeam sonar is the main sensor for ASV data acquisition, shown in Fig.6.2b. The sonar is integrated with the latest GNSS-assisted inertial navigation system (Applanix SurfMaster), has 80kH bandwidth, roll stabilization, an Ethernet interface and integrated sound velocity measurement. The basic sonar features are 5-210 degrees swath, adjustable measurement sector, 10mm resolution, 256-512 beams, 200kHz-700kHz nominal frequency 400kHz, range 0.2-275m (160m typical

at 400kHz). Ping rate up to 60Hz or adaptive, resolution: longitudinal x transverse standard 0.9x1.9degrees at 400kHz and 0.5x1.0degrees at 700kHz.

6.4 ASV Korkyra

The autonomous surface vehicle named Korkyra (see Fig.6.3) was developed as part of the HEKTOR project [114], [115]. It was designed as a catamaran to provide better stability and hydrodynamic properties in sea states up to state two. It is made of aluminum, 2000 mm long, 1000 mm wide, and has hollow hulls with a diameter of 240 mm. The upper deck consists of a carbon hull that houses all the electronics and computers that enable the autonomy of the vehicle. The catamaran's lower deck houses IP67-rated watertight aluminum boxes that house additional batteries, motor electronics, the NORBIT iWBMSc multibeam sonar system INS, and an expansion box that provides easy plug-and-play power, Ethernet and USB connectivity for each payload. The sonar, which is used in bathymetric applications of the ASV, is mounted in the front of the ASV on a bracket that can be lowered or raised as needed. The roll bar above the upper deck allows integration of maritime signal lights, surveillance camera(s), possibly even LiDAR, etc.

It currently weighs 100 kg in the air. Mobility of the ASV on land and during deployment from shore is provided by two rugged wheels on the rear of the hulls and a swivel wheel on the front. The ASV's stability and maneuverability at sea is further enhanced by 300 mm high keels at the front of each hull. On land, these keels act as front legs on which the ASV stands, and they also protect the payload mounted beneath the ASV.

The ASV Korkyra is equipped with various payloads, such as a multibeam sonar, a remotely operated vehicle (ROV), a tether management system (TMS) for the ROV, a landing platform for the LAAR, a pan-tilt-zoom (PTZ) Hikvision IP camera (mounted on top of the roll bar), and a LiDAR on the upper and front lower parts of the ASV. It can accommodate an additional 50 kg (in the air) of payload so that the hulls are semi-submerged for best hydrodynamic characteristics.

Four 390 W electric T200 thrusters in X configuration enable it to navigate complex marine environments at lower speeds (1-2 knots) in all directions. The thrusters are mounted on movable masts that can be adjusted in height. Thruster orientation in the horizontal plane can also be adjusted with a resolution of $45°$. In addition, the 720 W Minn Kota RT 55 EM booster electric motor enables top speeds of 3-4 knots. The booster motor is mounted in the rear of the ASV on the same height-adjustable bracket as the sonar. The total energy the ASV can draw from its batteries is 252 Ah or 3.73
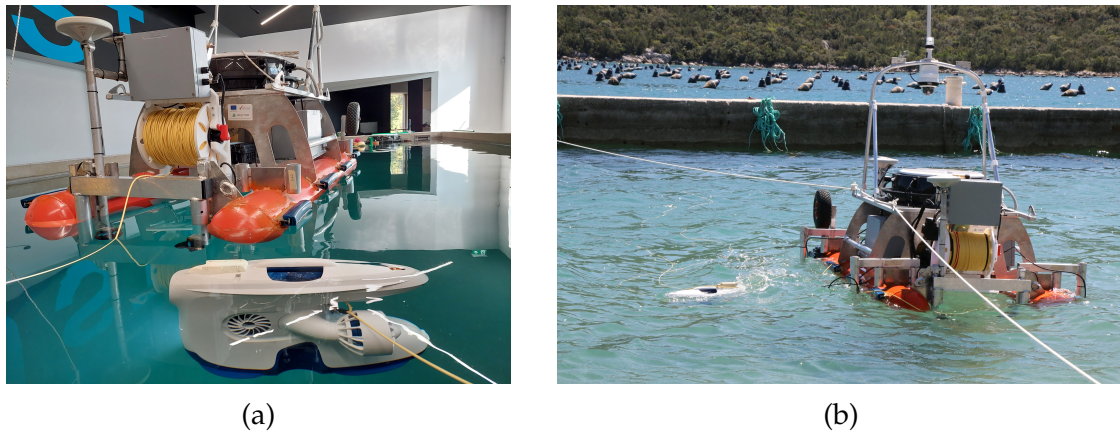
Figure 6.3: (a) ASV Korkyra with Blueye Pro ROV and TMS mounted in LABUST pool, Zagreb, Croatia. (b) TMS and underwater acoustical localization system tests at Bistrina Bay, Croatia, April 2022.

kWh. This means that the autonomy of the ASV Korkyra is between 3.5 h and 20 h. In mixed consumption scenarios, however, the ASV has an average autonomy of 10-11 h.

The GNSS with inertial navigation system IMU, named Applanix SurfMaster, combined with base station corrections over the Long-Term Evolution (LTE) network, enables the ASV to localize itself globally with an accuracy of up to 10 cm. The main computer uses ROS as a framework for mid- and high-level control, data processing and mission control. Communication with the vehicle is via Wi-Fi (Ubiquity Bullet M2 and an omnidirectional antenna on both the ASV and operator side) with a peak transmission speed of 100 Mbps over 400-500 m range. The option with Ubiquity Rocket M2 and Ubiquity $120°$ sector antenna was also tested to achieve longer range and better bandwidth. Operator work and mission planning for ASV Korkyra will be facilitated by open-source, graphical user interface-based software called Neptus.

6.5 Hybrid AUV/ROV robotic plaform e-URoPe

The hybrid AUV/ROV robotic platform e-URoPe (e-Underwater Robotic Pet) is an unmanned marine robot developed by the CNR-ISSIA, and it is characterized by reduced dimensions, 1.0 (length) x 0.7 (width) x 0.5 (height) m, and maximum operating depth of 200 m, see Fig.6.4. New design methodologies and material choice have led to the construction of a robust and highly reliable vehicle, as well as reduced weight and size turning into lighter logistics constraints. The vehicle guarantees complete navigation capabilities thanks to fully actuated propulsion configuration (4 horizontal and 4 vertical thrusters) and the presence of inertial sensors for attitude and acceleration measurements, combined with a DVL system for velocity reading. The integration of an USBL device provides the relative position localization for a more accurate navigation

Figure 6.4: e-URoPe hybrid AUV/ROV robotic platform used in the experiments.

capability during coordinated maneuvers. The exploitation of the optic fiber link (in ROV mode) allows the transmission of high-bandwidth sensor data (as cameras and multibeam sonar systems) as well as online functionality verification and debugging.

Chapter 7

Case studies

7.1 Introduction

This chapter presents a collection of case studies performed during numerous sea trials on a timescale of three years from 2017 to 2019. Section 7.2 brings experimental validation results of the MPC-based line following guidance module presented in Section 2.2 with simulation results. Underwater navigation is still a challenging problem in underwater robotics. Although this issues is out of the scope if this thesis, an Extended Kalman filter-based underwater navigation improvement module, mentioned first in Section 2.1 is here presented in Section 7.3. In Section 7.4, multibeam dataset collection trials at Plitvice Lakes National Park in Croatia were used for getting to know sonar mapping missions from operator's perspective and get ideas about autonomous coverage planning algorithms presented in Chapter 3. Sea trials during which a significant sonar dataset was collected at various underwater archaeological sites all around the Mediterranean are described in Section 7.5. This dataset is then used in Section 7.6 to validate algorithms for anomalous object detection in side-scan sonar imagery, previously introduced in Chapter 4.

7.2 Model predictive-based line following

7.2.1 Experiment setup

During the beginning of October 2016 we have conducted series of on-sea trials in Biograd na moru to validate simulation results. The hybrid AUV/ROV robotic platform e-URoPe (e-Underwater Robotic Pet, shown in Fig. 6.4) developed at Consiglio Nazionale delle Ricerche (CNR) (Genoa, Italy) has been used for the experiments. Since e-URoPe, which has been used in ROV mode, is an inherently fully actuated vehicle,

sway, pitch, and roll controllers had to be shut down. This way, it has been artificially transformed into an under-actuated vehicle, controlling only heave speed, surge speed, and yaw rate. The existing heave controller, already developed for e-URoPe, has been used to keep a constant depth. Experimental results validating the simulation results are given in Subsections 7.2.2 and 7.2.3 for surface and underwater experiments, respectively.

Firstly, the ACADO-ROS stand-alone simulator which was already developed needed to be integrated with the ROS environment which was developed by CNR for the e-URoPe ROV. The idea was to use MPC controller as yaw rate reference generator, while the e-URoPe simulator would be used for tracking of constant surge speed reference, and the MPC-generated yaw rate reference. Controllers of surge speed and yaw rate are PID controllers which have already been proven to have very good performance. Secondly, this integrated dual simulator structure needed to be connected with the e-URoPe ROV itself for the experiments to take place.

The line following model used in the experiments consisted of only states d, and β, given by Eqs. (2.9) and (2.10), respectively. The reason for excluding (2.11) from the model for the real system implementation was the increased oscillatory behavior of the vehicle when it gets close to the line due to integral action. This was the consequence of Global Positioning System (GPS) and Ultra Short Base-Line (USBL) measurement noise switching the vehicle's position from one side of the line to another. In order to minimize the d_{int} state, optimization algorithm tended to change the sign of the state d, resulting in unwanted oscillations of the vehicle around the lines of the lawnmower pattern. This simplified approach gave much better control performance in the experimental results compared to the previously used approach. One of the reasons for this is that in the simulations, the perfect knowledge of vehicle's position, surge speed, and yaw rate was assumed, so this case of measurement error conditioning the control performance could not have happened.

Since the d_{int} state was not used anymore, we have increased the K_β coefficient to $K_\beta = 0.1$ in order for the control optimization algorithm to put more effort into aligning the vehicle's heading with the orientation of the line being followed. Victory radius was increased to $R_{victory} = 2m$. Surge speed reference was set to a value $U_r = 0.2m/s$, since the vehicle is more maneuverable when moving at a slower speed. Also, the bounds for the yaw rate were lowered to $|r| \leq 12°/s$. Larger values of yaw rate bounds in the experiments have caused the controller to generate more aggressive control values near the bounds due to the position estimation errors, which in turn resulted in the unwanted oscillations of the vehicle around the reference line.

Experiments testing the performance of the developed MPC-PID framework have

been conducted both on the sea surface and underwater (at constant depth of $1.4m$). Low level yaw rate controller was a P controller with $K_p = 0.4$.

Also, both surface and underwater experiments were conducted with a guidance scheme from [116]. It is composed of a Lyapunov-based virtual-target path-following algorithm that generates a heading reference, combined with a PID heading controller which directly generated a yaw torque command. We used this method to compare the performance of the proposed MPC-PID framework with it.

7.2.2 Surface experiments

The first set of experiments has been conducted with e-URoPe ROV on the sea surface, using GPS for localization. In Fig.7.1a path of the ROV is shown when MPC-PID (red), and PID controller (blue) have been used. In both cases, the ROV shows oscillatory behavior when moving in the close vicinity of the given lines. MPC-PID controlled ROV converges to the line in a matter of $50s$ and keeps on moving really close to the line (see Fig.7.1b).

However, its heading error w.r.t. the desired line does not converge to a constant value, see Fig.7.1c. This value should be zero in case no disturbance is present, and a nonzero value in case disturbance is present and rejected. Aforementioned oscillations have an amplitude of less than $0.5m$, which is a small value of the same order of magnitude as the GPS sensor precision class. But in case that the ROV or any other autonomous marine vehicle with this control algorithm on-board are deployed for sea floor sonar scanning, the resulting interlaced sonar scans would be of greatly deteriorated quality.

In the Fig.7.1d it can be noted that the used low level yaw rate controller tracks the reference yaw rate values with perhaps only a small delay which is tolerable. This means that the subpar performance of the yaw rate tracking controller can be ruled out as a possible cause of the vehicle's oscillations around the given lines.

Oscillatory behavior could have been the consequence of the disturbances such as: waves hitting the GPS sensor causing measurement error (and also swaying the vehicle from the desired line), sea current, or unoptimized parameters of Kalman filter for position estimation. Also, it could be the case that the flat square-shaped front side of the ROV was causing it problems with movement on the sea surface due to the hydrodynamic effects, increased drag, etc.

(a) Reference lawnmower lines at a constant depth (black), defined by given waypoints (green markers). Vehicle's path while following the lines of the set lawnmower pattern: MPC controller (red), PID controller (blue). Heading of the vehicle controlled by MPC (black triangles).

(b) MPC controller. Distance to the given lawnmower line(s) (blue. Zero reference for the distance (red dashed).
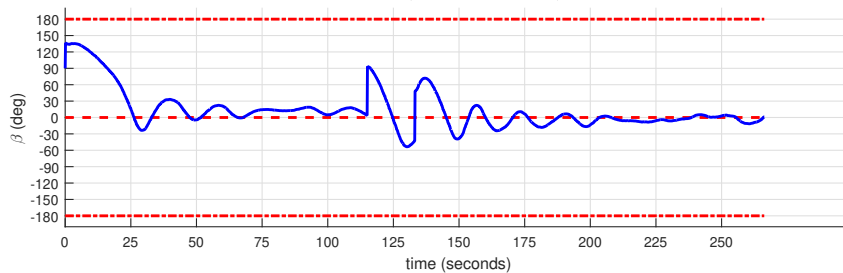
(c) MPC controller. Vehicle's heading error w.r.t. the orientation of the lawnmower line(s) being followed (blue). Zero reference for the heading error (red dashed). Bounds of the heading error values (red dash-dot).

(d) MPC controller. Estimated tracking (red) of the reference yaw rate (blue). Bounds of the yaw rate values (red dash-dot).

Figure 7.1: Surface experiments.

7.2.3 Underwater experiments

In addition to the experiments on the sea surface with e-URoPe ROV, underwater experiments were also conducted. During these experiments, USBL was used for localization. For the same given lawnmower pattern segment as in the Subsection 7.2.2, the path of the vehicle is given in Fig.7.2a, for both MPC-PID framework (red) and PID controller (blue). Compared to the line following on the surface, these underwater experiment results are much better. The vehicle does not oscillate around the line (except for a few position estimation outliers) in the case of both MPC-PID and PID controller. The distance to the desired line for MPC-PID framework (see Fig.7.2b) converges to zero in about $25s$, and stays close to zero.

Heading error w.r.t. the desired line orientation (see Fig.7.2c) also converges to zero as soon as the vehicle steadily converges to the line, around $25s$ from the start of the experiment. Since the experiments have been conducted in shallow waters, the effect of sea current was negligible, so the vehicle has been able to follow the desired lines, decreasing both the distance and the heading error to zero.

The same P controller mentioned in Subsection 7.2.2 has been used for yaw rate tracking, whose tracking performance for the underwater experiment is shown in Fig.7.2d. It can be noted that its performance is excellent, and that it has a short response time, meaning that the reference is tracked fast and precise. Also, reference yaw rate drops to value closer to zero as soon as the vehicle converges to the line, meaning that the vehicle moves with a constant heading, aligned with the followed line.

Looking only at the Fig.7.1a and 7.2a, and comparing the performance of MPC-PID framework and PID line following controller alone, it can be deduced that there is not a big difference between the control performance of the two approaches. However, if yaw rate tracking graphs in Fig.7.1d and 7.2d are analyzed, it can be noted that both the reference and accomplished values of yaw rate are within the set bounds.

On the other hand, when the guidance scheme from [116] was used, estimated yaw rate values violated the set bounds, which is shown in Fig.7.3. In this case, yaw rate was not controlled directly, so it could not even be saturated. Even if a PID yaw rate controller were used with a saturation block, this kind of control signal cut-off can, generally speaking, endanger the system's stability and deteriorate system's performance. Also, this kind of constraints violation on control signal can make control allocation problem much harder.

(a) Reference lawnmower lines at a constant depth (black), defined by given waypoints (green markers). Vehicles path while following the lines of the set lawnmower pattern: MPC controller (red), PID controller (blue). Heading of the vehicle controlled by MPC (black triangles).

(b) MPC controller. Distance to the given lawnmower line(s) (blue. Zero reference for the distance (red dashed).
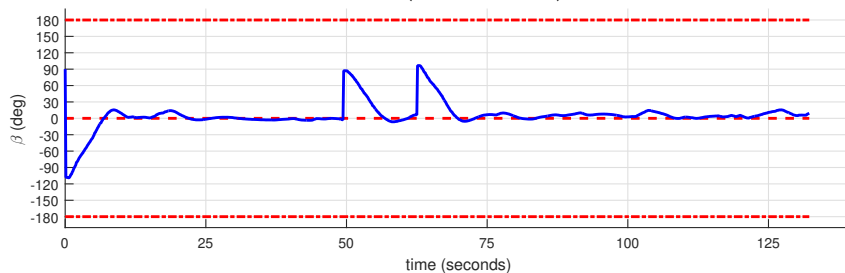
(c) MPC controller. Vehicle's heading error w.r.t. the orientation of the lawnmower line(s) being followed (blue). Zero reference for the heading error (red dashed). Bounds of the heading error values (red dash-dot).

(d) MPC controller. Estimated tracking (red) of the reference yaw rate (blue). Bounds of the yaw rate values (red dash-dot).

Figure 7.2: Underwater experiments.
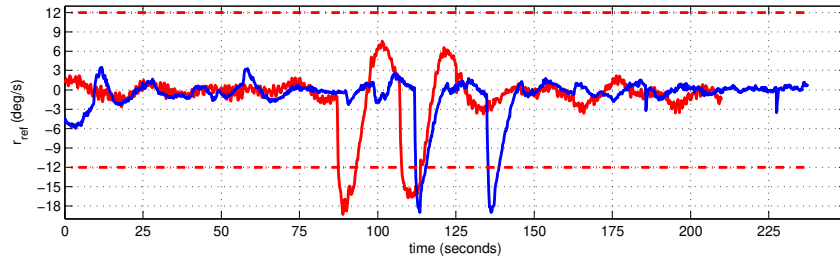
Figure 7.3: Estimated yaw rate signals caused by the use of PID controller. Surface experiment (red), underwater experiment (blue). Bounds of the yaw rate values (red dash-dot).

7.2.4 Conclusion

Real-time MPC framework implementation has been achieved for the line following problem which has been addressed in this section. Experimental results show good performance of MPC controller when compared to PID controller. The main advantage of MPC over PID control in this perspective is its possibility to be used as a general control optimization framework. Parts of this chapter were previously published by the author in [7,8].

7.3 AUV localization improvement

7.3.1 Introduction

In an underwater environment, Global Navigation Satellite System (GNSS) signals, that are available and widely used for localization in numerous land and air applications, are absent due to the very weak propagation of electromagnetic waves through the water. An electromagnetic homing system, presented in [117], is an example of the alternative that can provide accurate measurement of the autonomous vehicle position and orientation to the dock during homing, but also with very limited range due to same propagation constraints. Therefore, in the underwater environment acoustic based localization techniques are predominantly used [3].

Underwater navigation systems can be categorized in three categories, [118]. First, inertial navigation systems which use accelerometers and gyroscopes for increased accuracy to propagate the current vehicle state. Methods in this category have position error growth that is unbounded. Second, external acoustic systems are based on measuring the time–of–flight of signals from acoustic beacons or modems to perform navigation with bounded position error. Finally, geophysical navigation category contains techniques that use external environmental information as references for navigation. This is done with sensors and processing that are capable of detecting, identifying,
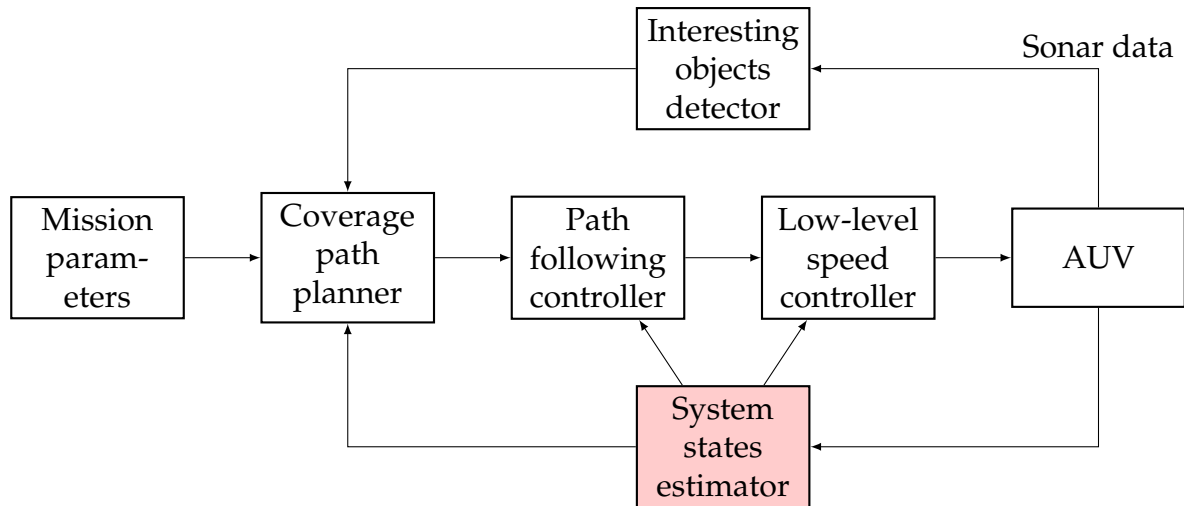
Figure 7.4: Block scheme of the entire AUV system.

and classifying some environmental features, e.g. cameras, sonars. Beside these three categories, when talking about localization we need to distinguish between two cases. In one case we want to determine vehicle position aboard the vehicle, while in second case, outside observer wants to determine vehicle position [3].

Since GPS is not available for UUVs' localization when underwater, it is possible to use various other sensors in order to improve the localization accuracy. One possibility is to augment UUV's localization by using a USBL alongside Inertial Measurement Unit (IMU), Doppler velocity logger (DVL), a Pressure sensor (PS), and GPS when available, with Simultaneous localization and mapping (SLAM) used for ground truthing, as presented in [119]. Other approaches use light beacons for close-range localization, as shown in [120], cooperative localization of a UUV assisted by a Unmanned Surface Vehicle (USV), as in [121] and [122], and a plethora of other sensors. A comprehensive overview of localization methods for UUVs is given in [123].

In order to achieve the above-mentioned goal, the first objective is to obtain a data set during sea trials involving one UUV and several USVs, and thus gather measurements from various sensors used to localize the UUV. The objective is to improve the existing localization of the UUV by augmenting its Extended Kalman Filter (EKF) localization filter in DUNE [124] with ground truthing UUV's position by detecting it one USV's downward looking camera's view, and in another USV's multibeam sonar images, as well as using range and bearing measurements of the USBL mounted below the USV, as shown in Fig.7.5a. The implementation and testing of the augmented localization filter were done in the post-processing phase of the sea trials, in order to see how much it enhances the accuracy of the existing UUV localization filter. The block scheme showing the overall view of the system with EKF as a "System states estimator" (in red) is given in Fig.7.4.

In real–life conditions, measurements provided by sensors are not ideal. They are noisy, may contain outliers, they are often delayed or intermittent. Some of them cannot even be directly measured. Due to that, measurements are rarely used directly in control systems. Instead, they are first estimated using some of many available filters. Also, mathematical models in practice are mostly nonlinear. Several nonlinear estimation techniques exist, some of which are the particle filter, unscented and extended Kalman filter. Particle filtering is a brute-force statistical estimator offering superior performance to Kalman filters for highly nonlinear systems. Unscented Kalman Filter (UKF) applies unscented transformations providing more accuracy than linearization, especially when propagating means and covariances. However, both the particle filter and UKF require more computation resources than the extended Kalman Filter, [125]. Nonlinear extensions of the Kalman filter use model linearization around the operating point. The most famous and used extension, using a first order linearization, is Extended Kalman Filter. Therefore, main equations are presented in this subsection for overview while complete EKF derivations can be found in [125,126]. Let a discrete nonlinear system be described with the following set of equations

$$\mathbf{x}_k = \mathbf{f}_{k-1}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right) \tag{7.1}$$

$$\mathbf{y}_k = \mathbf{h}_k\left(\mathbf{x}_k, \mathbf{v}_k\right) \tag{7.2}$$

$$\mathbf{w}_k \sim \left(0, \mathbf{Q}_k\right) \tag{7.3}$$

$$\mathbf{v}_k \sim \left(0, \mathbf{R}_k\right) \tag{7.4}$$

where $\boldsymbol{x}_k$ are system states, $\boldsymbol{y}_k$ outputs, and $\boldsymbol{u}_k$ inputs. Vectors $\boldsymbol{w}_k$ i $\boldsymbol{v}_k$ represent process and measurement noise described as Gaussian white noise with covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{Q}_k$, respectively. The main difference between a Kalman filter and its extended version is in the linearization of nonlinear $\boldsymbol{f}$ and $\boldsymbol{h}$ functions around the current state estimate. The general EKF algorithm is then summarized as, [127]:

1. Initialize the filter with:

$$\hat{\mathbf{x}} = \mathbf{E}(\mathbf{x}_0) \tag{7.5}$$

$$\mathbf{P}_0^+ = \mathbf{E}\left(\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)^\mathsf{T}\right) \tag{7.6}$$

where $\mathbf{E}(\cdot)$ is a expectation operator and $\boldsymbol{P}$ is the estimation error covariance matrix.

2. For each time-step k:

(a)Compute partial derivatives of the state equation:

$$F_{k-1} = \left.\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1}^+} \tag{7.7}$$

$$\mathbf{L}_{k-1} = \left.\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{w}}\right|_{\hat{\mathbf{x}}_{k-1}^+} \tag{7.8}$$

(b)Update the state estimate and covariance (*prediction*):

$$\hat{\mathbf{x}}_k^- = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}^+, \mathbf{u}_{k-1}, 0) \tag{7.9}$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{L}_{k-1}\mathbf{Q}_{k-1}\mathbf{L}_{k-1}^T \tag{7.10}$$

(c)Compute partial derivatives of the output equation:

$$H_k = \left.\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k^-} \tag{7.11}$$

$$\mathbf{M}_k = \left.\frac{\partial \mathbf{h}_k}{\partial \mathbf{v}}\right|_{\hat{\mathbf{x}}_k^-} \tag{7.12}$$

(d)Update the state estimate and covariance using the measurement innovation (*correction*):

$$K_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1} \tag{7.13}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left[\mathbf{y}_k - \mathbf{h}_k(\hat{\mathbf{x}}_k^-, 0)\right] \tag{7.14}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^- \tag{7.15}$$

where $\mathbf{K_k}$ is known as the Kalman filter gain.

EKF is commonly used in practice since the systems we want to estimate are non-linear. Due to model linearization around the operating point EKF, which introduces approximation error, is not an optimal estimator. If the initial state of the filter is chosen poorly, or the process model has large errors, filter can easily diverge. Despite that EKF gives satisfying performance, and presents de–facto standard in navigation systems. Marine vehicles are equipped with different sensors which provide measurements at different update rates. Hence, the navigation filter has a task to fuse available measurements, and by using the mathematical models provide state estimates at an update rate required by the control system.

Estimators which use dynamic models have several benefits over kinematic models. First, velocities are modelled exactly and generalized assumptions are not re-

quired. Kinematic models update their velocity estimate with position measurements which are often slow and infrequent, but command inputs to the kinetic model (X, Y, Z, N) are continuously available for velocity updates. Therefore, use of dynamic models enables separate kinematic and kinetic controller design uniformly across a wide range of vehicles and sensor suites, [5]. When dynamic model parameters are available, the model can be used as velocity measurement input into the kinematic model. However, combining the dynamic and kinematic model is more compact. The combined model inputs are generalized forces and moments which propagate the estimator at the controller update rate.

This Section is organized as follows: data collection sea trials and equipment used are described in Section 7.3.2, after which the preprocessing filters of the gathered sensor data are given in Section 7.3.3. The design of the EKF implemented in this section is presented in Section 7.3.4, and its navigational accuracy is compared to the existing EKF on-board the UUV in Section 7.3.4. Section 7.3.5 provides localization accuracy analysis of the implemented Extended Rauch-Tung-Striebel Smoother (ERTS) which was run at each UUV resurfacing. Key results of this section are concluded in Section 7.3.6.

7.3.2 Sea trials and data collection

The sea trials took place in the Degaussing Station in the vicinity of the Lora naval base in Split, Croatia, on 23-27 October 2017. The layout of the conducted experiments is shown in Fig. 7.5a. The UUV was planned to dive to around $5m$ depth (in order to still be visible in the USV's camera at that depth) and traverse a $200m$ long linear path. After each linear path, the UUV resurfaces, and returns to the start of the set path, doing this in total 5 times.

Torpedo shaped UUV Lupis (see Fig. 7.5b) controllable in the surge, pitch, and yaw was used for the experiments. When underwater, its onboard localization filter relies only on dead reckoning, i.e. on IMU, compass, DVL, and PS measurements for localization, while GPS measurements are not available. If available, i.e. the UUV is at the surface, GPS fixes are also integrated into the UUV's localization filter.

Overactuated USV Proteus [128] (see Fig. 7.5b) was mounted with a downward looking Logitech C920 HD Pro webcam, and a SeaTrac X150 USBL Beacon, shown in Fig. 7.5c. It was put in a Direct path (DP) mode approx. in the middle of the UUV's diving path, and oriented its bow towards the North, see Fig. 7.5a.

Due to the second USV shipping problems Teledyne BlueView M450 Series 2D multibeam imaging sonar could not be mounted onto the USV as planned. Instead, it was mounted on a lightweight aluminum tubular platform (see Fig. 7.5d) going $5m$
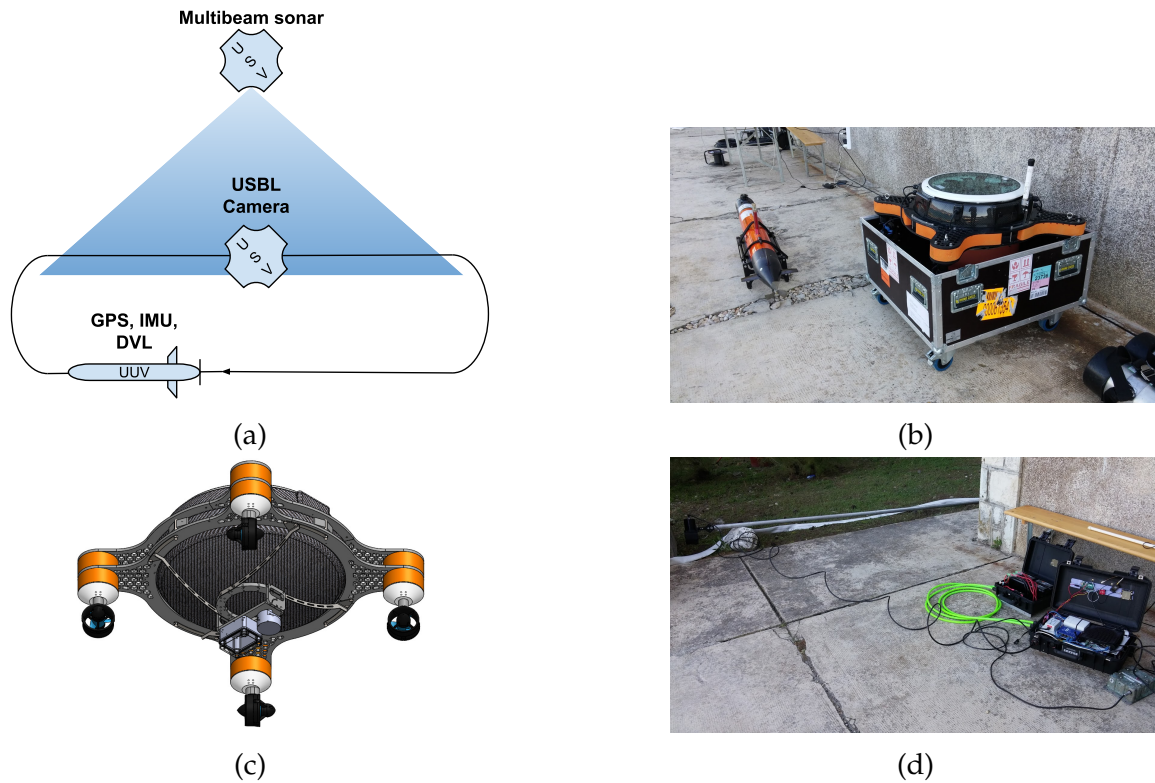
Figure 7.5: Experiments setup. (a) The layout of the experiments. (b) UUV Lupis and USV Proteus. (c) 3D model of the USV Proteus camera and USBL mount. (d) BlueView multibeam sonar and its mount pole (left), battery pack and processing unit (right).

into the sea, which was fixed to one of the pillars in the Degaussian station's waters, looking straight to the North, see Fig.7.5a.

7.3.3 Preprocessing of the Measurements

After finishing the sea trials, datasets of visual, sonar, and USBL data were processed. A simple online UUV visual detection algorithm was implemented. Also, the UUV was manually detected in multibeam sonar data, logging its range and bearing w.r.t. the sonar position. Range measurements of the USBL were shown to be useful while bearing measurements were not reliable and were almost completely random.

Surge speed outlier rejection filter

Due to DVL measurement noise, UUV's onboard EKF surge speed estimation had sudden and significant changes, even going from a positive to a negative value, which is physically infeasible. Surge speed outlier rejection filter is implemented in order to solve this problem and provide the EKF with feasible measurements. For a current surge speed measurement $u(k)$, its previous value $u(k-1)$, and the reference surge speed $\bar{u}$ (which was kept constant in each mission), surge speed is filtered in

a few steps. First, it goes through a saturation block with limits $u_{min} = 0[m/s]$ and $u_{max} = 2[m/s]$, and its output is denoted by $u^s(k)$.

After that the difference of the surge speed $u^s(k)$ and its expected reference mean value $\overline{u}$ is computed by $\delta^{\overline{u}}_{u^s}(k) = |u^s(k) - \overline{u}|$, by which $u^s(k)$ filtering is then conditioned with:

$$u^*(k) = \begin{cases} u(k-1) + \text{sgn}(\delta^{u(k-1)}_{u^s(k)})\delta^{\overline{u}}_{u^s}(k)/\delta_p, \delta^{\overline{u}}_{u^s}(k) > \delta_{th} \\ u^s(k), otherwise \end{cases} \tag{7.16}$$

where $\delta^{u(k-1)}_{u^s(k)}$ is given by $\delta^{u(k-1)}_{u^s(k)} = |u^s(k) - u(k-1)|$, $\delta_p = 25$ is the predefined filter parameter that represents the part of $\delta^{\overline{u}}_{u^s}$ for which $u^s(k)$ can change in one discrete time step, and $\delta_{th} = 0.3[m/s]$ is the predefined filter parameter that represents the threshold of $u^s(k)$ change w.r.t. its constant reference value $\overline{u}$ in one discrete time step.

Results of surge speed estimation outlier rejection are given in Fig.7.6a. It can be noted that numerous spikes in the range of samples $250-600$ are successfully smoothed by the filter. It is even more important that the filter copes well with sudden false negative values of estimated surge speed, which are present in samples $800-900$. In these cases, the filter maintains the value of filtered surge speed around the expected reference value $\overline{u}$.

Visual detection filter

In order to detect the UUV in the USV's downward looking camera's view, a simple visual detection algorithm is implemented. It counts the number of pixels in a frame which has the hue in some range close to the calibrated hue of the UUV at the operating depth. Good performance of both false positives and negatives rejection was achieved by empirically based thresholding of the characteristic hue range pixels count, as well as their outspread. centre of gravity (CoG) of all detected pixels in the set hue range is set to be the ground truth UUV position measurement.

An example of the UUV caught in camera's view is shown in Fig.7.6b, while the estimated output UUV CoG position is shown in Fig.7.6c. The hull of the UUV is approximated by the red lined minimum bounding box, while the red dot represents UUV's estimated position. It can be noted that the UUV visual detector works very well. Equations of simple frame transformations from camera frame coordinates to local NED frame coordinates $(x_{cam}(k), y_{cam}(k))$ were omitted here for brevity.

Sonar image based detection filter

Detection of the UUV in multibeam sonar images was done manually, logging range r^{uuv}_{son} and bearing β^{uuv}_{son} of the UUV w.r.t. to the sonar position. Since side-scan sonar on-

board the UUV was active throughout the mission, significant interferences are visible in multibeam sonar's imaging. In order to quickly get range and bearing measurements, no automatic detection method was used, because it would need significant amount of fine-tuning to prevent false positives and negatives. After the detection phase, the polar coordinates range r_{son}^{uuv} and bearing β_{son}^{uuv}, with a known sonar location $(son_{N_{abs}}, son_{E_{abs}})$ in the local North-East-Down (NED) frame, were transformed into the absolute UUV position measurements (x_{son}, y_{son}). An example of the UUV detection in the sonar image is given in Fig.7.6d.

USBL range outlier rejection filter

In order to reduce USBL range measurement outliers, a simple range limiting filter is used before the USBL's range measurement r_{usbl}^{uuv} is fed into the augmented EKF. The basic idea is to accept the range measurement only if it is feasible taking into account vehicle's surge speed and time difference between two consecutive range measurements. If the time difference of the two consecutive USBL measurements is no more than $\delta_t^{usbl} = 3s$, then the range is filtered by:

$$r_{usbl}^{uuv}(k) \leftarrow \begin{cases} r_{usbl}^{uuv}(k-1) + \text{sgn}(\delta_r^{usbl})\delta_{r,max}^{usbl}, \delta_r^{usbl} > \delta_{r,max}^{usbl}, \\ r_{usbl}^{uuv}(k), otherwise \end{cases} \tag{7.17}$$

where $\delta_r^{usbl} = |r_{usbl}^{uuv}(k) - r_{usbl}^{uuv}(k-1)|$ denotes the absolute change in range of two consecutive measurements, and $\delta_{r,max}^{usbl} = \delta_t^{usbl}\overline{u}$ denotes the maximum feasible value of δ_r^{usbl}. Filtering of the USBL's range measurements is shown in Fig.7.6e, for all 5 line segments which the UUV traversed underwater. It can be noted that the raw range measurements do not have many outliers, but the above-presented filter successfully rejects even the relatively small number of range outliers.

7.3.4 Extended Kalman Filter Used for Sensor Fusion

Process Model

A six degrees of freedom (6DoF) discretized kinematic model of the underactuated rudder-steered UUV is used as the process model of the EKF, with state vector $\eta(k) = [x(k), y(k), \psi(k), u(k), v(k), r(k)]^T$. where x, y and ψ denote x-, y- coordinates of the UUV in the local NED frame, and its heading, respectively. Surge, sway, and yaw velocities are denoted by u, v and r, respectively.

The vertical component of the UUV's position is neglected since the mission design ensured that the vehicle moves at a constant depth plane during its linear path fol-
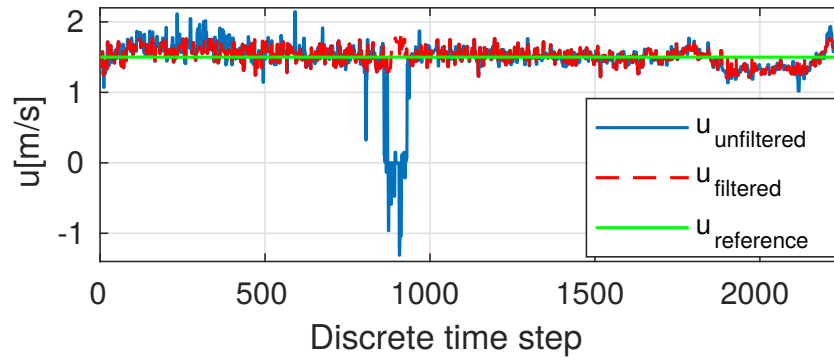
(a) An example of the implemented surge speed outlier rejecting filter output.
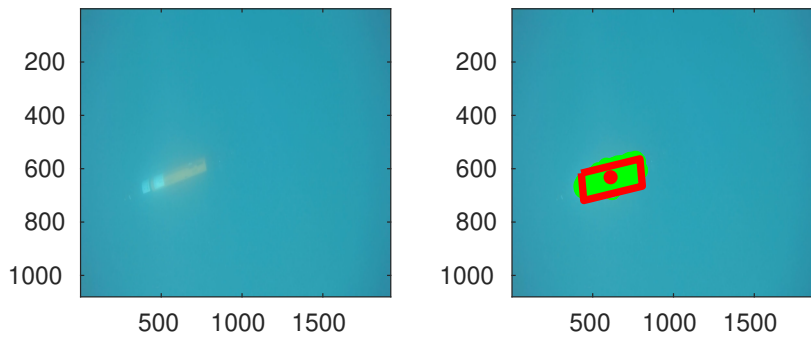
(b) An original frame from USV's downward looking camera capturing the UUV in its view.

(c) An example of the automatic UUV detection in the downward looking camera frame.
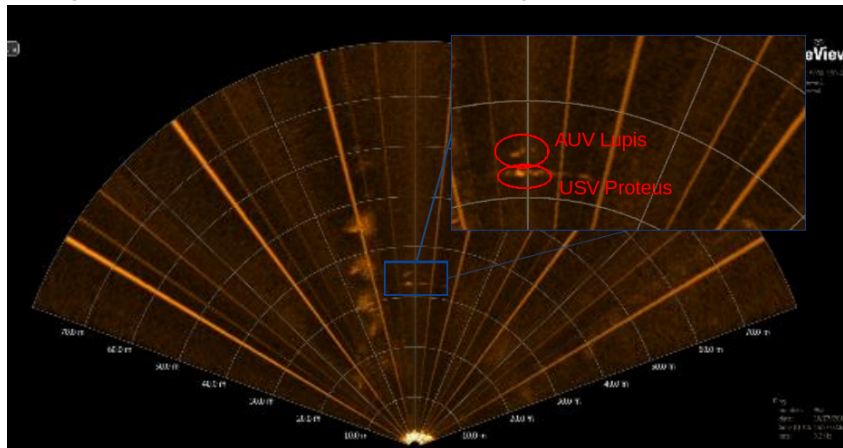
(d) An example of manual UUV detection in a sonar image.
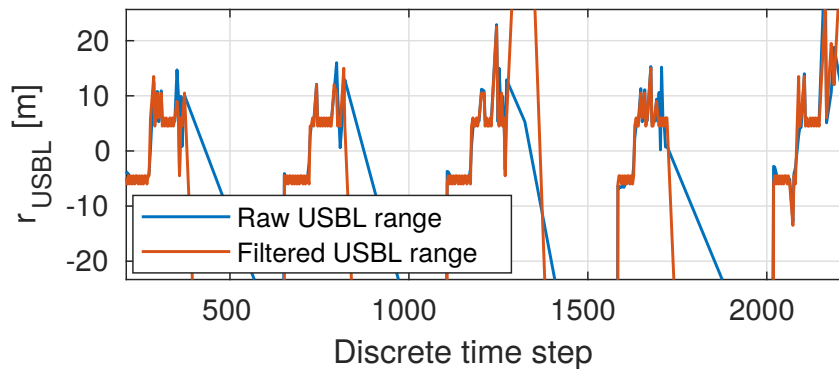
(e) An example of filtering out USBL range measurements outliers.

Figure 7.6: Examples of the surge speed, camera, sonar, and USBL measurement preprocessing.

lowing. Also, the 3DOF (x, y, ψ) process model is extended with zero dynamics surge, sway, and yaw velocities model. This way, the perturbation of the velocities' values is allowed through process noise parameters of the EKF.

Process noise covariance matrix Q is designed as a diagonal matrix with elements: $q_{11} = q_x = 0.01$, $q_{22} = q_y = 0.01$, $q_{33} = q_\psi = 0.001$, $q_{44} = q_u = 0.001$, $q_{55} = q_v = 0.01$, and $q_{66} = q_r = 0.1$, which are taken from the existing localization filter of the Lupis UUV, in its DUNE navigation layer [124].

Measurement Model

The measurement model of the EKF implemented in this section is made to be modular, i.e. it gets augmented whenever a new type of measurement is available. Basic localization uses dead reckoning to localize the UUV, and it is being improved when GPS, and/or camera, and/or sonar, and/or USBL measurements are available.

Dead reckoning module   Basic localization of the UUV when underwater is dead reckoning, so measurement vector z_m contains only surge, sway, and yaw velocities measurements, taken as the velocities estimates from the existing UUV's EKF filter $z_m(k) = [u(k), v(k), r(k)]^T$, and measurement noise covariance matrix $R = \text{diag}(\rho)$ is a matrix with its diagonal elements taken from vector ρ, $\rho_1 = \rho_u = 1e{-}3$, $\rho_2 = \rho_v = 1e{-}4$, and $\rho_3 = \rho_r = 1e{-}3$, which were taken from the existing localization filter of the Lupis UUV, in its DUNE navigation layer [124]. Measurement model values h vector is given by $h(k) = [\hat{u}(k{-}1|k{-}1), \hat{v}(k{-}1|k{-}1), \hat{r}(k{-}1|k{-}1)]^T$, where $\hat{u}(k{-}1|k{-}1), \hat{v}(k{-}1|k{-}1)$, and $\hat{r}(k{-}1|k{-}1)$ are the estimates of states u, v, and r from the previous EKF iteration.

GPS localization module   In case that the UUV is at the surface, GPS fix measurements are available, so the dead reckoning measurement model is augmented with the GPS fix coordinates and the corresponding estimated GPS variance. The measurement vector z_m is thus extended with the GPS fix latitude and longitude values converted into the UUV's local NED frame $z_m(k) \leftarrow [z_m(k), x_{GPS}(k), y_{GPS}(k)]^T$, and the measurement noise covariance matrix R diagonal is extended by two additional diagonal elements, i.e. $\rho(k) \leftarrow [\rho(k), \rho_x^{GPS}(k), \rho_y^{GPS}(k)]^T$, where $\rho_x^{GPS}(k) = 0.1$, and $\rho_y^{GPS}(k) = 0.1$. The measurement model values h vector is extended as $h(k) \leftarrow [h(k), \hat{x}(k{-}1|k{-}1), \hat{y}(k{-}1|k{-}1)]^T$, where $\hat{x}(k{-}1|k{-}1)$, and $\hat{y}(k{-}1|k{-}1)$ are estimates of states x, and y from the previous EKF iteration.

Visual localization module   In case that the UUV is detected in the USV's downward-looking camera, its position is calculated by the visual detection filter presented in

Subsection $7.3.3$. The measurement vector z_m is thus extended as $z_m(k) \leftarrow [z_m(k), x_{cam}(k), y_{cam}(k)]^T$, and the measurement noise covariance matrix R diagonal is extended by two additional diagonal elements, i.e. $\rho(k) \leftarrow [\rho(k), \rho_x^{cam}(k), \rho_y^{cam}(k)]^T$, where $\rho_x^{cam}(k) = 0.1$, and $\rho_y^{cam}(k) = 0.1$. Measurement model values h vector is again extended with $\hat{x}(k-1|k-1)$ and $\hat{y}(k-1|k-1)$ state estimates.

Sonar localization module   If the UUV is detected in the multibeam imaging sonar, its position is calculated by the sonar image-based detection filter described in Subsection $7.3.3$. The measurement vector z_m is thus extended as $z_m(k) \leftarrow [z_m(k), x_{son}(k), y_{son}(k)]^T$, and the measurement noise covariance matrix R diagonal is extended by two additional diagonal elements, i.e. $\rho(k) \leftarrow [\rho(k), \rho_x^{son}(k), \rho_y^{son}(k)]^T$, where $\rho_x^{son}(k) = 0.25$, and $\rho_y^{son}(k) = 0.25$. The measurement model values h vector is extended again with $\hat{x}(k-1|k-1)$ and $\hat{y}(k-1|k-1)$ state estimates.

USBL localization module   As soon as the UUV dives, USBL fix from one of the USVs becomes available, and UUV's position is estimated only based on the range measurements from the USBL, as presented in Subsection $7.3.3$. The measurement vector z_m is thus extended as $z_m(k) \leftarrow [z_m(k), r_{USBL}(k)]^T$, and the measurement noise covariance matrix R diagonal is extended by an additional diagonal element, i.e. $\rho(k) \leftarrow [\rho(k), \rho_r^{USBL}(k)]^T$, where $\rho_r^{USBL}(k) = 0.1$. The measurement model values h vector is extended with the state $\hat{r}_{USBL}(k-1|k-1)$ where $\hat{r}_{USBL}(k-1|k-1)$ is the modeled range measurement that is equal to the Euclidean distance d_{UUV}^{USV} between the UUV and the USV mounted with the USBL, i.e. $\hat{r}_{USBL}(k-1|k-1) = d_{UUV}^{USV}(k-1|k-1)$.

Results

The comparison of localization performance of the implemented EKFs with different measurements available is given in Fig. 7.7 for one of the 5 linear paths that the UUV tracked underwater. It can be noted that with each additional measurement added to the filter, the better its performance is, i.e. the closer the UUV's estimated position is to the first stable GPS fix when resurfacing.

The comparison of x- and y-position covariance values for the EKF with different measurements available is given in Fig. 7.8. It can be noted that camera-aided localization helps reduce the estimated uncertainty only in a few samples when the UUV is visually detected. Sonar's range is on an order of magnitude wider than camera's, so it reduces the estimated uncertainty for a much longer time. Finally, the USBL range
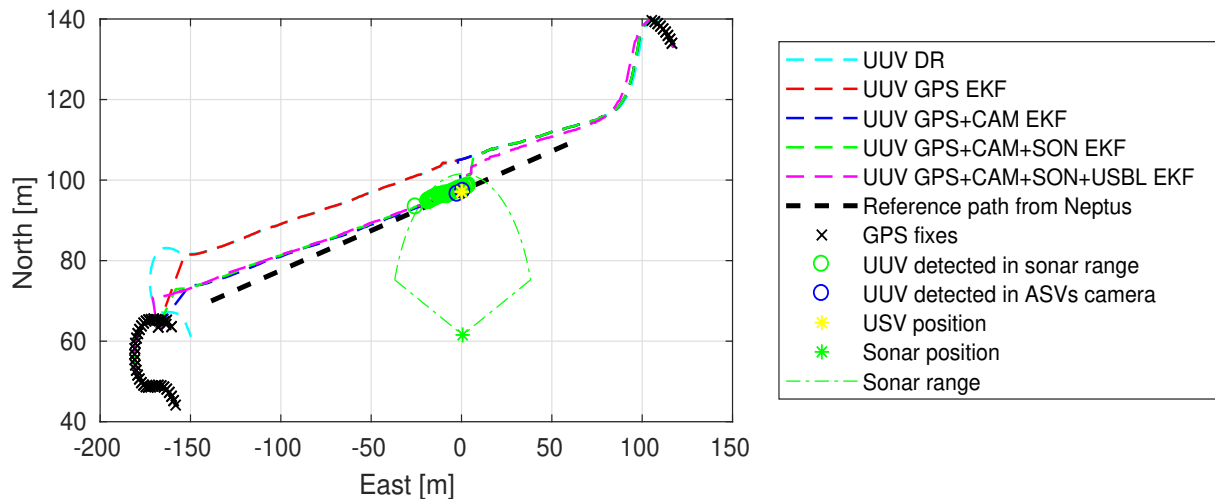
Figure 7.7: An example of the implemented EKF localization of the UUV with different measurements available. Start of the UUV traversing the transect is in the top right, while the end of the transect is at the bottom left end of the transect.

is the longest, so it starts reducing UUV's position estimate uncertainty as soon as the UUV dives.

Table 7.1 shows the aggregated comparison of all EKF approaches which use different measurements for UUV's localization improvement, for all 5 transects which the UUV traversed in a racetrack mission. The performance by which the EKF approaches are compared is determined at the moment when the UUV resurfaces, as the distance of the estimated UUV's position from the first stable GPS fix. It can be noted that with the gradual augmentation of the EKF its performance becomes better. It is interesting to note that adding range-only USBL measurements improved the performance of the EKF in 40% of the cases. However, when comparing camera, sonar, and USBL-augmented to dead reckoning (DR) localization, and EKF with GPS fixes, the improvement of the proposed approach is $40-69\%$, and $33-65\%$, respectively, which represents a significant improvement.

7.3.5 Extended Rauch-Tung-Striebel Smoother

In applications in which it is needed to have higher precision localization a-posteriori, e.g. in photogrammetry, bathymetry applications etc, it is useful to use a smoother in the post-processing phase of the camera, sonar or other data acquisition. In this case, after applying EKF filtering forward in time for each of the line paths, an Extended Rauch-Tung-Striebel (ERTS) smoother of the first order was applied backward in time in each resurfacing time instant. This yielded navigation uncertainty minimization and consequently improvement of the smoothed localization w.r.t. the filtered localization. The ERTS is used after each EKF with different measurements available, mentioned in
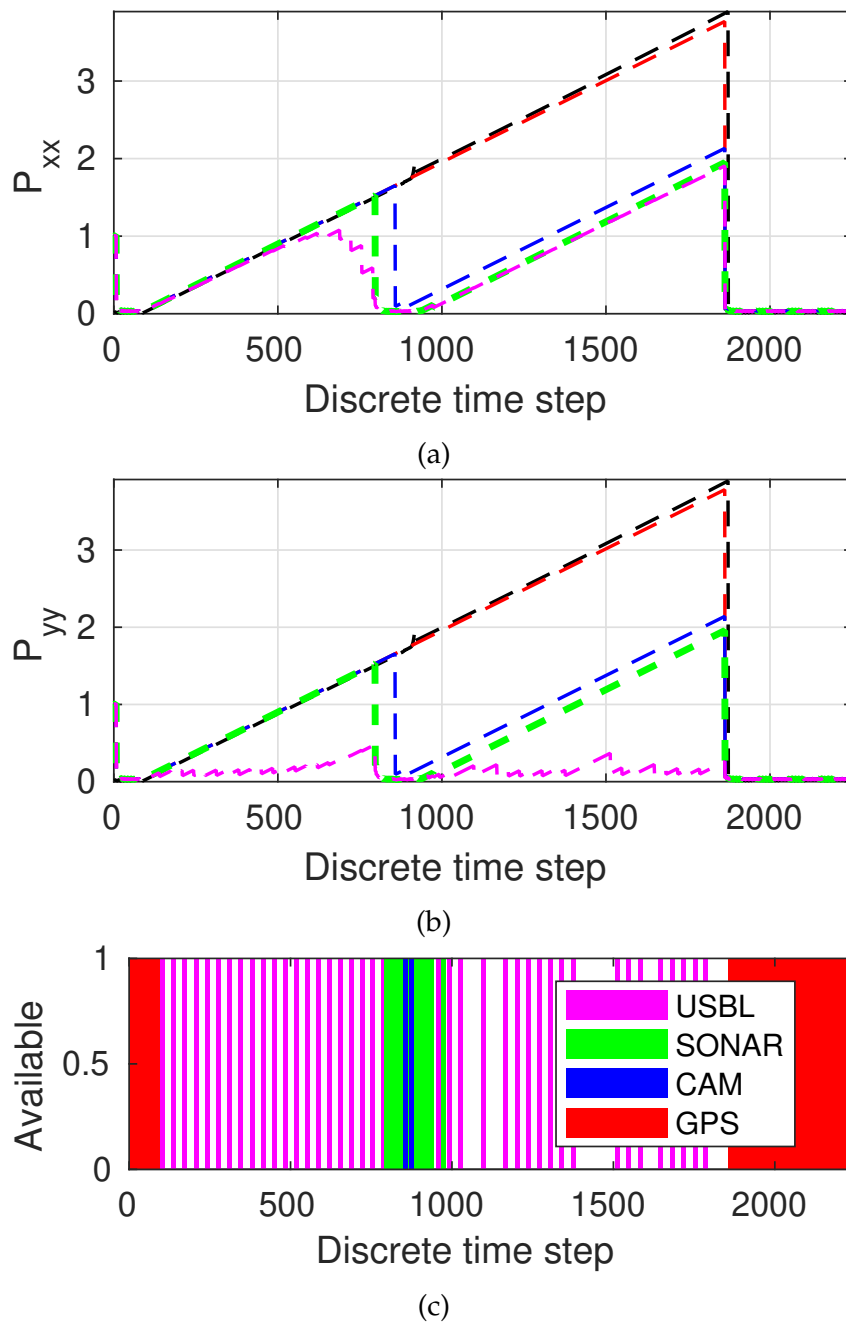
Figure 7.8: An example of localization uncertainties of the implemented EKF w.r.t. the available measurements. The evolution of (a) the covariance P_{xx}, for North component of UUV's position in the local NED frame, (b) the covariance P_{yy}, for East component of UUV's position, and (c) measurement availability. For legend, the reader should refer to Fig.7.8c. The black dashed line represents navigation uncertainty of the UUV's EKF from DUNE [124].
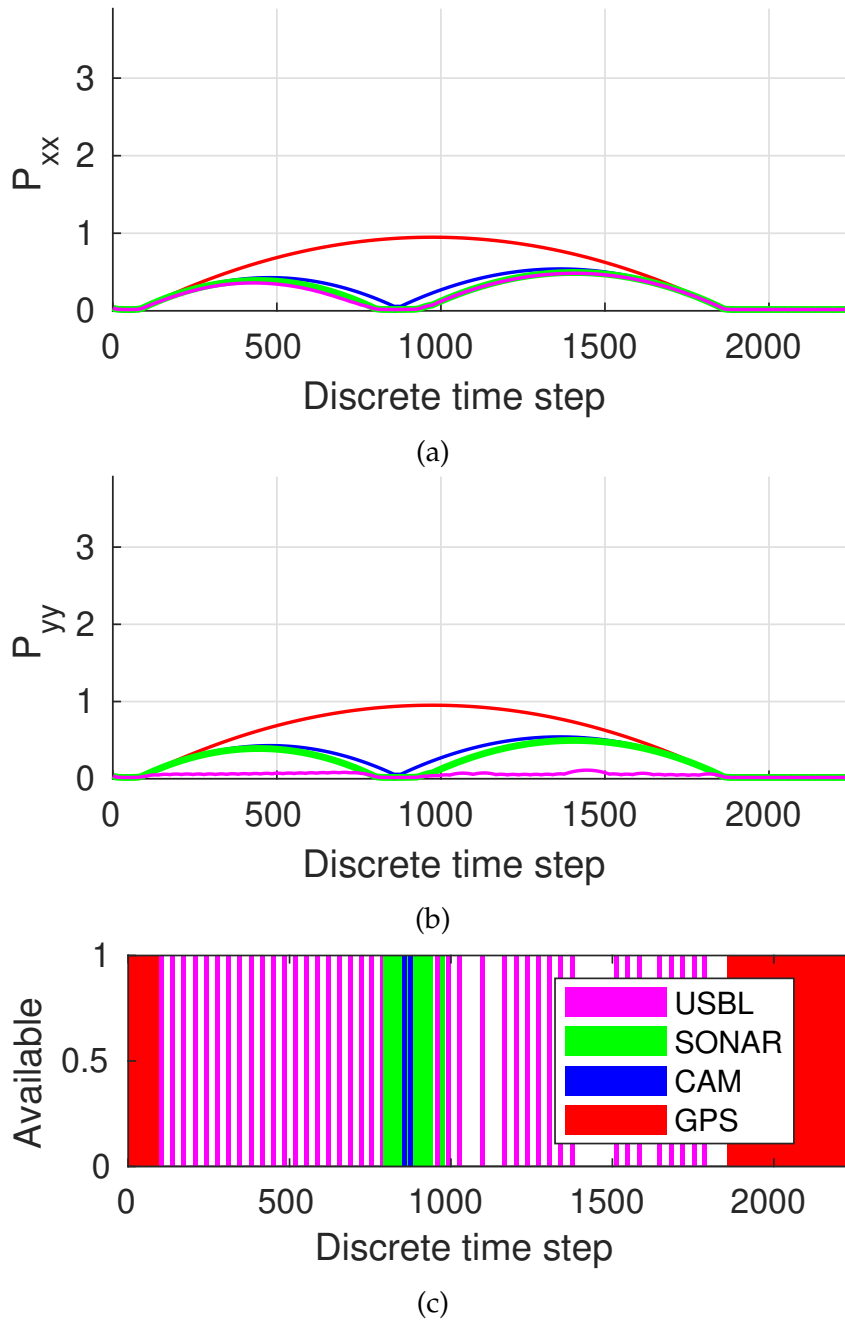
Figure 7.9: An example of localization uncertainties of the implemented ERTS w.r.t. the available measurements. The evolution of (a) the covariance P_{xx}, for North component of UUV's position in the local NED frame, (b) the covariance P_{yy}, for East component of UUV's position, and (c) measurement availability. For legend, the reader should refer to Fig.7.9c.

Table 7.1: Aggregated performance of the EKF localization improvement for 5 transects each of which was $200m$ long. Performance measure: the distance of the estimated UUV's position from the first stable GPS fix while resurfacing.

	#1	#2	#3	#4	#5
DR	27.15	28.29	25.99	33.35	18.96
GPS EKF	24.76	24.00	23.11	31.28	18.75
GPS+CAM EKF	18.05	20.30	20.12	20.96	18.75
GPS+CAM+SON EKF	14.12	16.05	13.55	16.65	6.24
GPS+CAM+SON+USBL EKF	13.89	16.07	8.10	17.57	11.25

the previous section.

The performance measure used to compare four implemented ERTS smoothers is given by:

$$J = \sum_{k=1}^{N} |d(k)| P_{xx}^{ERTS}(k) P_{yy}^{ERTS}(k) \pi T_s \tag{7.18}$$

where $d(k)$ is the Euclidean distance of the UUV from the set transect to follow, the product $P_{xx}^{ERTS} P_{yy}^{ERTS} \pi$ is the area of the navigational uncertainty ellipsis, and T_s denotes sampling time. This way, the computed distance of the vehicle to the given line is weighted by the level of certainty that the smoother has about the accuracy of the distance value.

Results

The comparison of x- and y-position covariance values for ERTSes with different measurements available is given in Fig.7.9. It can be noted that the ERTS smoother drastically reduces the localization uncertainty when compared to EKF given in Fig.7.8. This is of course expected since the smoother has full information about the previous measurements when propagating back in time.

Table7.2presents the aggregated results of ERTS performance in various scenarios of different measurements' availability. It can be noted that the more measurements are available, the vehicle is estimated to follow the line better with its smoothed path. In this case, the ERTS augmented with GPS and camera is more than twice more accurate than the ERTS based only on GPS measurements. In case #5 there were no detections of the UUV in the camera, so GPS and GPS+camera ERTS performances are the same. Smoother which is augmented by the in-sonar UUV position measurements is about 25% more accurate than the ERTS using only GPS and camera. Lastly, ERTS further

Table 7.2: Aggregated performance (scaled by 1000) of the ERTS smoothed localization improvement for 5 transects each of which was $200m$ long.

	#1	#2	#3	#4	#5
GPS ERTS	4.91	4.85	4.60	3.27	5.85
GPS+CAM ERTS	1.89	1.73	1.78	1.48	5.85
GPS+CAM+SON ERTS	1.51	1.36	1.45	1.19	1.70
GPS+CAM+SON+USBL ERTS	0.19	0.20	0.23	0.16	0.37

augmented with USBL measurements is approximately 6 times more accurate than the ERTS using GPS, camera, and sonar for UUV localization. This presents a significant improvement of the UUV localization with each added measurement type.

7.3.6 Conclusion

A modular measurement model EKF for UUV underwater localization is proposed. Except for using measurements from UUV's sensors, namely compass, IMU, DVL, PS, and GPS, this EKF is augmented by USBL range and visual-data based localization from a USV, and in-sonar image estimated UUV position. It is shown that the proposed EKF significantly enhances UUV's navigational accuracy through a collaborative fusion of sensor data from multiple heterogeneous marine vehicles. Also, ERTS smoother was run a-posteriori, to test how much it can improve UUV's localization for post-processing of the data acquired by the UUV (e.g. camera, side-scan sonar, multibeam sonar etc). It was shown that additional measurements of UUV's position can significantly improve line following accuracy. Parts of this section were published by the author in [4,129].

7.4 Plitvice lakes dataset acquisition

7.4.1 Introduction

Plitvice Lakes National Park (cro. Nacionalni Park Plitvička jezera) is the oldest and largest national park in the Republic of Croatia. The park is located in the mountainous region of Croatia, as the map in Fig.7.10 shows. With its exceptional natural beauty, this area has always attracted nature lovers, and as early as 8 April 1949 it was declared the first national park in Croatia. Dominik Vukasović, the parson of the Plitvice Lakes
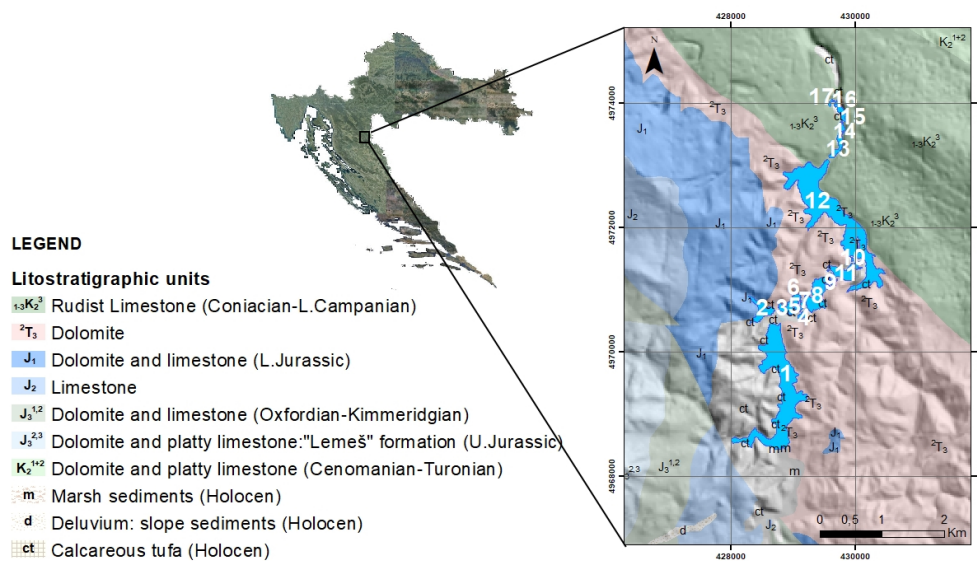
Figure 7.10: A simplified geological map of the Plitvice National Park modified after [133]. The shape of the lakes and the surrounding terrain closely reflect the underlying geological structure. The bedrock of the Upper Lakes (denoted 1-12) is mainly the Upper Triassic dolomite that is relatively impermeable and retain water, making Upper lakes resting in gently sloping valley. The bedrock of the Lower Lakes (dentoed 13-16) is the Upper Cretaceous limestone that was susceptible to karstification processes, making Lower lakes narrower and situated in deep canyon. Courtesy of Josip Barbača from Croatian Geological Survey.

area in 1777, was the first to mention the name "Plitvička jezera". The name is most likely derived from the Croatian word for shallow water (pličina, or plitvak), [130,131].

The process of tufa formation, which results in the building of the tufa, or travertine barriers and thus to the creation of the lakes, is the outstanding universal value for which Plitvice Lakes were internationally recognized on 26 October 1979 when they were inscribed on the UNESCO World Heritage List UNESCO. In 1997, the boundaries of the National Park were extended, and today it covers an area of almost $300 km^2$, [132].

The lake system consists of 16 named and several smaller unnamed lakes, which cascade into each other. Due to the geological substrate and the characteristic hydrogeological conditions, the lake system was divided into the Upper and Lower lakes. The twelve lakes that form Upper Lakes were formed on impermeable dolomite rock and are larger, with more articulated and gentler shores than the lower lakes. The Lower lakes were formed in permeable limestone substrate cut into a deep canyon with steep cliffs. The lakes end in the impressive Sastavci waterfalls, with the Korana River springing from the base of the falls.

Depth measurements and the characterization/classification of the lake bottom are of utmost importance for research of tufa formation and eutrophication processes in the lakes. Major Franz (Franjo) Bach was the first to carry out depth measurements of

Plitvice Lakes in 1850, [130]. His work was improved by contributions from Franić in 1910, [131], Gavazzi in 1919, [134], and Petrik in 1958, [135]. All these studies involved measuring depths directly by a rock tied to a wire, which was then thrown from a small boat at as many points as possible. These measurements were very valuable for their time, but they were not georeferenced and the depth measurements were much less accurate compared to modern indirect depth measurement methods.

It is only in the last 20 to 30 years that we have been able to explore and map the underwater world of the Earth, mainly through technical advances such as acoustic remote sensing. Models based on acoustic data can be used to estimate how underwater locations have changed both recently and far in the past, but also to predict how they might change in the future, [136,137]. These models can then be used as powerful tools in public commitment to environmental protection and conservation. Nothing has advanced in underwater technologies and research areas as much as localization and environmental imaging devices. Recently, photogrammetry, photo modeling, simultaneous localization and navigation (SLAM), organized light processing, multibeam and numerous other acoustic sensing techniques have become ubiquitous [138–141]. In 2007 and 2010 Pribičević and his colleagues bring modern acoustic methods of bathymetric measurements of the two largest lakes in Plitvice, namely Kozjak and Prošćansko lake, [142] and [143].

There are many understandable logistical limitations to working in Plitvice Lakes National Park. Due to the minimization of the carbon footprint that people leave behind in the National Park, any type of gasoline-powered vehicles are strictly prohibited, so only electrically powered vehicles come into play. Even access to the lakes with all the research equipment can sometimes be extremely strenuous. The morphology of the terrain is very rough and steep. Around some lakes there are forest paths only a few meters wide, while some lakes are crossed by quite narrow wooden footbridges. The installation of a multibeam sonar on a small boat with electric motor drive seems to be a simple but effective solution. However, the main problem with this human-operated surveying approach is that the planning and execution of surveying missions is prone to human error and could therefore result in some parts of the lake bottom not to be covered by the sonar.

7.4.2 Methodology

In March and April 2019 extensive bathymetric measurements were carried out on the lakes in Plitvice Lakes National Park in Croatia. The aim of the depth measurements of the lakes is to enable a detailed environmental monitoring of tufa formation and changes over time. The bathymetric measurements were performed by ASV PlaDy-

(a)   (b)

Figure 7.11: (a) Start of survey operations at lake Okrugljak. Two-person manual deployment of the ASV PlaDyBath from the shore of the lake. (b) Mission planning for the ASV PlaDyBath from a makeshift workstation.

Bath. In total, three of four of the Lower Lakes, and eight of twelve of the Upper Lakes were surveyed.

These lakes have never been surveyed with sonar technology, except the largest lakes Kozjak and Prošćansko jezero, whose bathymetric surveys were carried out by National Park and its partners some years ago, [142–144]. The remaining lakes were either too shallow for safe operation of the ASV PlaDyBath or the deployment of the vehicle was too complicated due to the terrain configuration. Processed sonar data was delivered in a form of digital elevation maps (DEMs) to National Park Plitvice for further analysis and research of the water column.

A high-resolution georeferenced orthophoto image based on the camera images of the unmanned aerial vehicle (UAV) was used as a basis for the planning of survey missions, as shown in Fig.7.12. The methodology to generate such a large-scale orthophoto is partly described in [145]. This model was calculated before the bathymetry survey missions to serve as a crucial input for the safety of the ASV PlaDyBath. The outlines of the lakes were extracted from these precisely georeferenced orthophoto models with an offset of about $5-10m$ from the shore to avoid shallow parts or other potentially dangerous obstacles along the shore, as shown in Fig.7.13a and Fig.7.14a. Particularly frequent obstacles were the tree trunks that fell into the lakes and were not moved by the National Park staff, as any human intervention in the National Park is strictly prohibited. It can also be noted that the outline of the exact lake boundary does not match the map background from the map loaded in Neptus, because the resolution of the Neptus map layer is not sufficient to recognize the boundaries of the lakes.

The missions were planned as follows: Plan the first mission along the perimeter of the lake, with the sonar beams tilted 30-60° towards the shore and the swath angle at
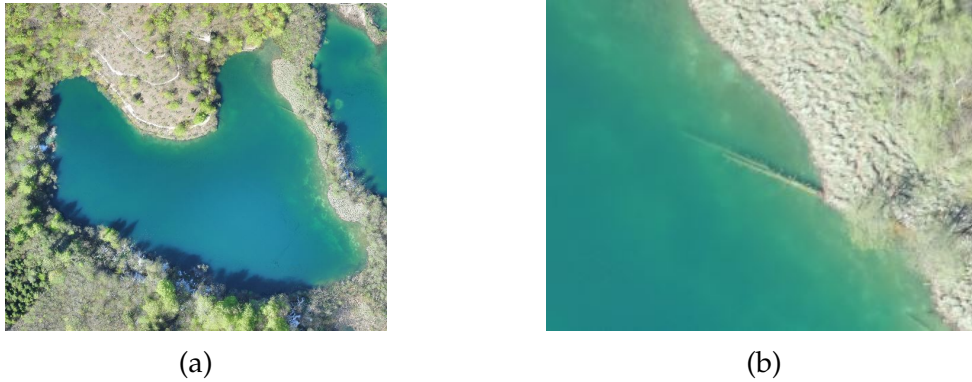
Figure 7.12: High resolution digital orthophoto image of the Okrugljak lake. (a) Orthophoto of the whole lake used for high precision georeferenced lake outline extraction. (b) Detail of a tree trunk sunk in the shallow waters of Okrugljak lake. Such areas were then excluded from the ASV's planned missions.

90-120°. The rest of the lake was covered with a sonar tilt of 0° (looking directly under the vehicle) and a swath angle of 90°. The ping rate of the sonar was set to adaptive in order to obtain as much quality data as possible. This meant that the sonar pinging frequency increased in shallow areas and was automatically reduced in deeper parts of the lake. The ASV's surge speed was set at 0.7 m/s, which has proven to be a good compromise between energy consumption and surveying time during a number of survey missions. Depending on how shallow/deep a lake is, the rest of the survey for this lake was carried out as follows.

When a lake had a more or less convex shape, as is the case with Malo jezero shown in Fig.7.13a., the interior of the lake was partially covered by the so-called lawn mowing maneuvers with variable lawn mower cross-section widths. In the shallowest parts of the lake denser transects were used, while in deeper parts of the lake wider transects were used, as shown in Fig.7.13b.

If the lake was deep or had a rather non-convex shape, as is the case with Lake Ciginovac (see Fig.7.14a), the interior of the lake was covered with concentric, bank-shaped missions, as shown in Fig.7.14b. For this type of non-automated mission planning, it is essential that the vehicle is in WiFi range throughout the time of each circumferential pass, in order to check the depth at each waypoint it passes. The inward offset of the next concentric mission from the waypoints of the current mission is calculated as $d = 2(1 - \alpha/2)h \tan(\psi/2)$, where d is the inward offset, h is the depth read by the WBMS software at the specific waypoint reached by the vehicle in the current mission, ψ is the sonar swath angle and $\alpha \in (0, 100\%]$ is the overlap percentage of adjacent sonar swaths. We have used $\alpha = 20\%$ and $\psi = 90°$, which gives $d = 1.8h$. We also smoothed sharp curves as much as possible to reduce the fan-out effects in the sonar footprint, which result in much sparser sonar pings on the outside and unnecessary

(a)   (b)
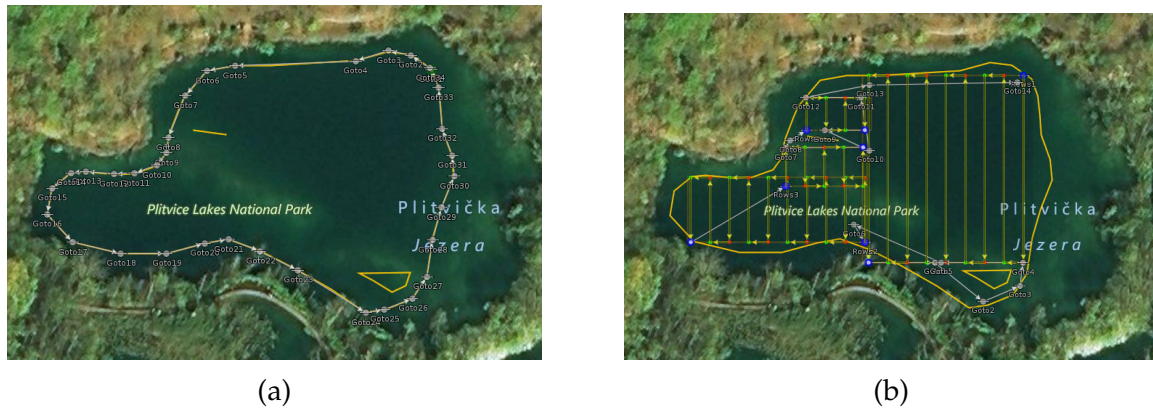
Figure 7.13: Background: Satellite imagery of Lake Malo jezero loaded in Neptus mission planning software for the ASV PlaDyBath. (a) The initial survey mission: The safety outline 5-10m from the lake shore imported from a shape file based on the high resolution orthophoto with a centimeter georeferencing precision. This ensured that the ASV does not get stuck in shallow water bellow $0.5m$ deep, or that it gets stuck into many tree trunks and branches which fell into the lake. Survey mission waypoints are placed along this outline. (b) The rest of the lake covered by lawnmower patterns of different widths depending on the estimated depth of the lake.

multiple overlap of the pings on the inside of the turn towards the center of the lake.

While PlaDyBath performs its tasks autonomously, the operator processes sonar data in low resolution using the Qimera software to monitor the quality of coverage. As soon as the batteries are depleted (approx. every $1.5 - 2h$), the vehicle was lifted to the lake shore, the batteries were changed, and high-quality sonar data were transferred to the computer of the ASV operator. Initially, the break in surveying operations was used not only for data transfer but also for passive cooling of the Applanix Surf-Master navigation system. During the $1.5 - 2h$ surveys on lakes, especially in shallow lakes where the adaptive pinging rate would increase significantly, it heated up very much. As mentioned before, the adaptive ping rate was used to get as much data as possible w.r.t. of the given water column depth, therefore it was not an option to lower it. After the first visit to Plitvice Lakes National Park a cooling system consisting of a small fan and a heat outlet was developed, which significantly improved the performance of the ASV during the second visit to Plitvice, as it solved the problem of overheating.

7.4.3   Results

Bathymetric data of all survey lakes were processed in the QPS Qimera software to create 2.5D manifolds representing interpolated depth profiles. The reconstruction parameters used for the reconstruction, based on sonar and IMU datasheets, were heading uncertainty: $0.3°$, roll and pitch uncertainty: $0.08°$, and the unit cell size for surface

(a)   (b)
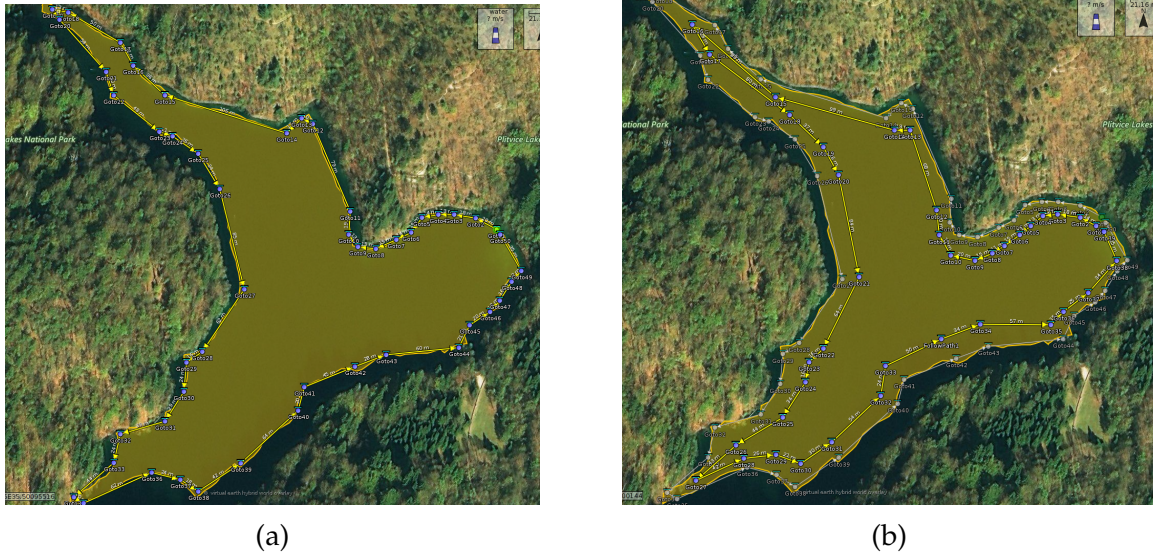
Figure 7.14: Background: Satellite imagery of Lake Ciginovac loaded in Neptus mission planning software for the ASV PlaDyBath. (a) The initial survey mission: The safety outline $5-10m$ from the lake shore imported from a shape file based on the high resolution orthophoto with a centimeter georeferencing precision. This ensured that the ASV does not get stuck in shallow water bellow $0.5m$ deep, or that it gets stuck into many tree trunks and branches which fell into the lake. Survey mission waypoints are placed along this outline. (b) The rest of the lake covered by concentric circumference missions. Each inner mission is offset inwards by $1.8 \times depth$ measured at the waypoints of its neighboring outer mission.

interpolation was set to $0.2m$.

Digital elevation maps (DEMs) extracted from the 2.5D bathymetry manifolds were then overlaid with the digital orthophoto of the area based on UAV camera images. Upper Lakes DEM is shown in Fig.7.15 and Lower Lakes' DEM is given in Fig.7.16. Note that the darker parts on the right side of the lakes are only the result of the standard shader in the Global Mapper software used to merge the bathymetric DEMs with photogrammetric orthophotos in Figs.7.15 and 7.16. It is interesting to see how the tufa-shaped lake bottom does not slope evenly from the shore to the middle of the lake. On the contrary, it contains many depressions, which are the result of the interaction between water and tufa bottom.

It is important to note that Petrik in [135] gathered in total 5400 limnological measurements of various types about all 16 lakes during a four-year period in the 1950s. The total number of data points processed (or so-called cells in the interpolated surfaces) in the QPS Qimera software is over 7.5 million for 13 lakes. This number is an order of magnitude lower than the number of raw sonar measurements. Collecting the measurements took 7 working days, and processing the sonar data took another 8 days, for a total of 15 days. This means that the state-of-the-art autonomous robotic systems, equipped with the latest visual and acoustic remote sensing technologies, en-
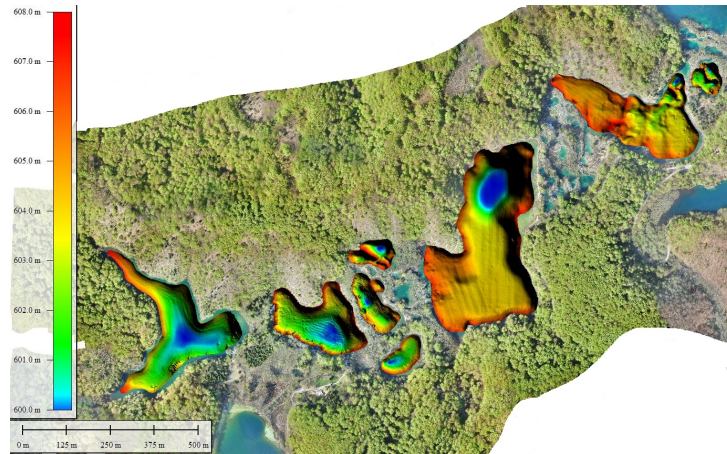
Figure 7.15: Upper Lakes' georeferenced bathymetric DEM layered over the digital orthophoto of the area. Top right to bottom left: Burget, Gradinsko, Galovac, Malo, Veliko, Burget, Ciginovac, and Okrugljak lakes.
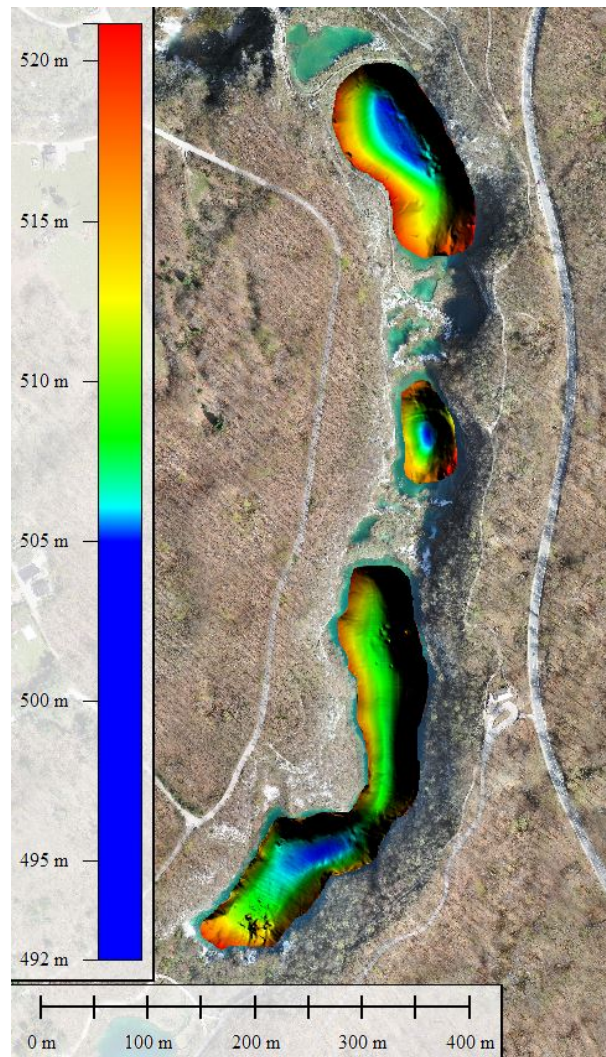
Figure 7.16: Lower Lakes' georeferenced bathymetric DEM layered over the digital orthophoto of the area. Top to bottom: Kaluđerovac, Gavanovac, and Milanovac lake.

abled the researchers to collect over 1400 times more bathymetric measurements in a time frame more than 60 times shorter than that of [135].

7.4.4 Conclusion

A great potential for the use of autonomous vehicles in remote sensing studies is presented in this section. Here an autonomous surface vehicle and an unmanned aerial vehicle were used for hydrology related autonomous remote sensing survey missions of Plitvice Lakes National Park in Croatia. The efficiency of using autonomous vehicles for such applications, the quality and quantity of data in time and the precision of georeferencing the data, especially in larger areas as in the authors' case study, has been shown to far exceed human performance. A total of 11 of 16 lakes in the National Park were surveyed using acoustic surveying methods with a multibeam sonar.

This section presents the scientific basis and methodology used in the modern autonomous robot geodetic measurements in the area of Plitvice Lakes National Park, its processing and the development of a digital three-dimensional geodetic models of lakes. These digital bathymetric maps will be the basis for further research purposes in GIS environment by experts in scientific disciplines such as biology, geology, hydrology, ecology, etc. Decisions on the way and form of protection of the underlying phenomenon of tufa formation in Plitvice lakes would be made on the basis of these state-of-the-art maps. Part of the results presented in this section are published in [146].

7.5 Underwater archaeology dataset acquisition

7.5.1 Intoduction

Sometimes new technologies take time to completely fulfill their potential uses. Hence, technology is often a bridge between different disciplines. Methods for recording and documenting underwater cultural heritage (UCH) sites have evolved significantly in the last two decades. The combined use of optical and acoustic technologies enables the provision of quality digital 3D reconstruction of large and complex underwater scenarios, [111,140]. These technologies create the opportunity to study the UCH in the laboratories onshore in a nonintrusive manner [147]. Resulting digital reconstructions are often accepted for archaeological purposes, and in particular for documentation and monitoring activities, [107,148]. These tools can be used to predict how sites have changed, both recently and far into the past but also how they may change in the future. These tangible results can be used as powerful aids in public engagement.

Nothing has advanced as significantly in underwater archaeology as location and environment imaging equipment has. Recently, photogrammetry, photo-modeling, simultaneous localization and navigation (SLAM), organized light processing, multibeam and numerous other acoustic remote sensing techniques were all used on Mediterranean underwater sites, [138–141]. However, even as archaeologists are eager to exchange the laborious manual documentation process for more effective processes, no particular system has shown enough straightforward advantages to be universally embraced or recognized as the current standard for digital site reporting. Price, precision, durability, and time post-processing problems are typically of utmost importance. Often a challenge for archeologists who usually lack the expertise to process the data themselves is the opportunity to incorporate multibeam sonar point clouds and photomosaics to generate archeologically relevant diagrams and publication-quality charts.

Marine robotics is evolving as a powerful remote sensing aid in shallow sea, providing a broad variety of pre-disturbance survey possibilities (2.5D modelling of a location or landscape without excavation). Marine robots are not confronted with the technical difficulty of activities in deep water in these coastal underwater archaeological environments, but are potentially met with a much greater obstacle when they reach direct contact with human divers. Archaeologist scuba divers incorporate high versatility, smart coordination, and a large variety of manual skills to reduce operational costs. However, these human advantages tend to vanish when the region to be explored gets larger, deeper, or the period required for field operations is shorter.

This section encompasses the case studies of activities regarding the use of the autonomous marine vehicles for recording the underwater archaeological sites. It brings in total eight case studies of marine robotics survey applications for UCH sites around the Mediterranean. Applied methodology for data collection is described for each individual site, since each one of them had its specifics. Depending on the site, the authors used all or some of the autonomous marine vehicles (ASV, AUV, ROV, UAV) for data collection. It is important to emphasize that most of the underwater archaeological sites presented in this section were for the first time recorded by autonomous marine vehicles, as well as by sonars. Part of the results presented in this section are published in [107,136,148–150].

The rest of this section is organized as follows: data acquisition and processing pipeline are described in Subsection 7.5.2. Underwater archaeological sites surveyed in the scope of BLUEMED project (2016-2019) are given in the following Subsections, namely Underwater archaeological park of Baiae (Italy) in Subsection 7.5.3, amphorae cage and ancient dolii sites close to Cavtat (Croatia) in Subsection 7.5.4, four ancient Greek shipwreck from the Western Pegaseticos in Subsection 7.5.5. Finally, the survey

results of S.M.S. Szent Istvan shipwreck (sunk in 1918) made in 2019 are presented in Subsection 7.5.6. Concluding remarks are given in Subsection 7.5.7.

7.5.2   Data acquisition and processing pipeline

Multibeam sonar data

When surveying with MBES one should be aware of the physical limitations of sonar beam resolution as well as the performance limit of spatial resolution of samples seafloor patches. The size of the "footprint" $\delta_b \times \delta_b$ which MBES beam sonifies for beam width angle α at depth h can be estimated as:

$$\delta_b = 2h \tan \frac{\alpha}{2}. \tag{7.19}$$

If we take that the mean across- and along-track beam with of Norbit WBMSc sonar is $1°$, we get $\delta_b = 0.02h$. Furthermore, spatial resolution of the MBES δ_s, i.e. the distance between adjacent sonar beams at the assumed flat seafloor, can be estimated as:

$$\delta_s = 2h \tan \frac{\psi}{2N} \tag{7.20}$$

for a given swath angle ψ, number of beams N and depth h. For a swath angle $\psi = 90°$ and $N = 256$ which were used in most of the surveys, we get $\delta_s = 0.006h$. We used these equations do determine the upper bound of the bathymetric model interpolation grid cell size.

The autonomous surface vehicle (ASV) PlaDyBath's survey path was designed lawnmower-shaped survey missions along and across the area of interest. In order to have as much quality coverage as possible, it is needed to overlap the adjacent lawnmower lanes with percentage $k \in (0, 1]$. The distance between the lawnmower lanes d for a given overlap percentage can be estimated for the assumed approximately flat seafloor as:

$$d = 2 \left(1 - \frac{k}{2} \right) h \tan \frac{\psi}{2}. \tag{7.21}$$

During the acquisition phase there were issues with swath width angle of $\psi = 120°$, especially at $700kHz$ frequency. This was due to the fact that the floater foams on port and starboard of the ASV blocked some sonar beams or the sonar was not submerged deep enough below the ASV. This was solved by using the $400kHz$ frequency for higher range but with reduced swath angle at $\psi = 90°$. With swath angle of $90°$ and overlap $k = 0.5$ the distance between the lawnmower lanes was easily computed as $d = 1.5h$.

Norbit's WBMS software was used for logging raw MBES data, both for the point

cloud and backscatter data. QPS Qimera was used for processing the raw sonar data. In order to process the data in Qimera, the .son format of raw sonar data decoupled from the navigation data had to be exported to .s7k format which was then loaded into Qimera. Applanix INS logged the navigation data into .pos format files that could be directly imported into Qimera. Based on Applanix datasheet, the orientation data uncertainty was set to $0.3°$ for heading and $0.08°$ for roll and pitch. The heading and roll/pitch precision was based on patch tests performed by the instructions given by Norbit. Precision of GPS position data was on the order of $10cm$ when only Trimble antennae were used connected to the Applanix INS. This precision was improved to the order of $1cm$ if NTrip client was connected through a 4G modem, when we got access to local base station system. Both GPS and INS data were used to merge sonar data with navigation data and generate the bathymetric model. The bathymetric models were georeferences in UTM system.

Raw sonar pings had to be filtered for outliers in Qimera. This was done by limiting the range of depth values appropriate for the location as was read from the WBMS software. Swath Editor was used to select across-track outliers for several tens of pings at the same time, while Slice Editor was used to filter our the pings on the outside of the delimited UCH area, as well as ping which were recorded while the ASV was turning from one lawnmower lane to the next. The turning motion causes the so-called fan-out effect, where the inside of the turning curve is sampled much more that the sparsely sample outside part of the turning path. This unevenly sampled data was not desired so it was deemed as outliers.

Interpolation was done using the default interpolant from Qimera. NOAA CUBE interpolation was also used, but it did not show better results. The grid cell size for the interpolation was based on calculations of δ_s, which was the lower limit for this parameter. Since Applanix INS does not have a heave sensor integrated, waves from the surface were translated directly to the bathymetric model. This was solved using Wobble Analysis tool in Qimera as well as additional spline interpolation for larger areas with fewer details.

Side-scan sonar data

Survey missions for the LAUV Lupis mounted with the side-scan sonar were planned in a form of the lawnmower pattern at constant altitude from the seafloor. Distance between adjacent lawnmower lanes was determined with the assuming that the seafloor is locally approximately flat, and that the maximum slant range is ten times larger than the operating altitude. Since all UCH sites except for one were at depth of $6-30m$ operating altitude was $2-3m$ for safety reasons so that the AUV does not collide with

the artifacts at the seafloor if the amphorae/anchor mounds rose too suddenly from the flat surrounding. Assuming locally flat seafloor the lawnmower lane width was set in the range $15 - 30m$. In some cases we got significant surface returns because the AUV was relatively close to the surface, especially at Baiae UCH site where the depth was $8m$ at most. The side-scan sonar was pinging at both low and high frequency, so we could detect the differences in the resolution and object detection methods. For the analysis we of course used only high frequency data.

We do not have direct access to the raw side-scan sonar data. Neptus Mission Review and Analysis software downloads the mission logs from the LAUV Lupis either over WiFi or from the USB located in the nose part of the vehicle. It then processes raw side-scan sonar data, merges it with the navigation estimation data, and presents it in the waterfall view. The position data represent the filtered dead reckoning estimation augmented with the DVL and GPS measurements when available. The path of the vehicle is corrected in the post-processing phase from the last stable GPS fix after resurfacing to the beginning of the dive. There is also an option for adjusting the time varying gain (TVG) as needed. Except for TVG, the brightness of side-scan sonar imagery can be equalized using Contrast Limited Adaptive Histogram Equalization algorithm (CLAHE) [151] as shown by the authors in [78]. Finally, side-scan sonar and all other navigation and control log data can be exported from Neptus in many formats: .kml, .mat, .csv just to name a few.

AUV localization precision is on the order of $0.5 - 1m$ since a now very precise GPS sensor gives the upper bound on localization precision. When underwater, the localization uncertainty increases significantly as dead reckoning only accumulates error, but is corrected by the DVL as soon as it gets bottom lock

Visual data

Visual surveys of some UCH sites and their surrounding both underwater (by BlueROV and/or AUV Lupis) and above sea surface (by DJI Phantom UAV). Camera onboard AUV Lupis has $1.4MP$ and is fixed to capture things directly below the vehicle, so it cannot be used for detailed full 3D modelling of a site. However, since it is synchronized with an external flash it can be used for creating orthophotos or photomosaics. Images from AUV Lupis have timestamps and can be correlated with its position estimation data to further simplify photo alignment of photogrammetric software. BlueROV was used for capturing oblique HD images of some UCH sites and creation of 3D photogrammetric models. It does not have any underwater localization system, not even dead reckoning, so its images were fed "as is" into photogrammetric software after batch automatic brightness filtering in Photoshop. UAV DJI Phantom captures

Figure 7.17: The location of the Baiae underwater archaeological site

images in $20MP$ resolution. It was used for capturing the shoreline in the vicinity of the UCH sites. It has a GPS sensor, so it embeds it into metadata of the images which photogrammetric software can read later on. The precision of UAV's GPS is $1-2m$.

Agisoft Metashape was the software used for photogrammetric processing of images captures by all three vehicles. Photo Alignment, Point Cloud generation and Mesh processes were all set to high settings to get as detailed models as possible given the quality of the input images. The use of underwater robots for photogrammetric recording purposes was a use case to prove how effective these systems can be compared to diver photographers, especially at depths of more than $50m$. Quality of the models is graded rather subjectively since there are no precise models which could be used as benchmarks.

7.5.3 Baiae site

Baiae, ancient city of Campania, Italy, is located on the west coast of the Puteoli Gulf (Pozzuoli) and lies 16 km west of Naples, as shown in Fig.7.17. Baiae was called after Ulysses' helmsman Baios according to custom. Because of its curative sulfur springs the city is known as Aquae Cumanae in 178 BC. The mild climate of Baiae, the thermal springs and luxuriant vegetation made it a popular resort during the Roman Republic's later years. Many splendid villas, including those of Julius Caesar and Nero, were established at Baiae. Owing to nearby seismic activity (bradyseism), more than 100 meters of the ancient site is now underwater in the harbor.

Methodology

In 2018, our team went to Baiae to record the underwater archaeological site at depths of $6-8m$. This was done under Interreg Mediterranean co-funded project BLUEMED.

Figure 7.18: Side-scan sonar imagery mosaic recorded by LAUV Lupis

Survey operations at Baiae pilot site were conducted by our team on 14-18.5.2018. The main emphasis of this operation was to gather as much bathymetric data of the site as possible, so the ASV PlaDyBath was mostly deployed for survey missions. For this reason, the ROV has not been used for visual inspection of the pilot site, and UAV has not been used since we did not have an official permission for operating it in the Baiae port area. The boundary of the survey area has been determined based on a GIS map provided by the local diving center. Detailed survey missions were planned based on the georeferenced mosaic of side-scan sonar imagery recorded by LAUV Lupis, see Fig.7.18.

A boat was rented from the local diving center for the bathymetry and photogrammetry data collecting operations by the surface and underwater autonomous marine vehicles. It was quite spacious so deployment and recovery of the autonomous vehicles by the crane was not problematic. The autonomous surface vehicle (ASV) PlaDyBath's survey path was designed lawnmower-shaped survey missions along and across the area of interest in the Baiae bay. The missions were planned with $90 - 120°$ field of view angle of the Norbit multibeam sonar used for bathymetry, having in mind to cover the whole area with complete overlap between any two adjacent along-track survey lines, and also having across-track survey lines to maximize the amount and quality of the bathymetry data as much as possible, and to avoid holes in the bathymetry map. As soon as one mission would finish, the sonar data and position/attitude data were transferred from the ASV to a laptop, and bathymetry data were processed. Meanwhile, another mission was started, so data collection and processing were parallelized as much as possible.

(a)

(b)

(c)
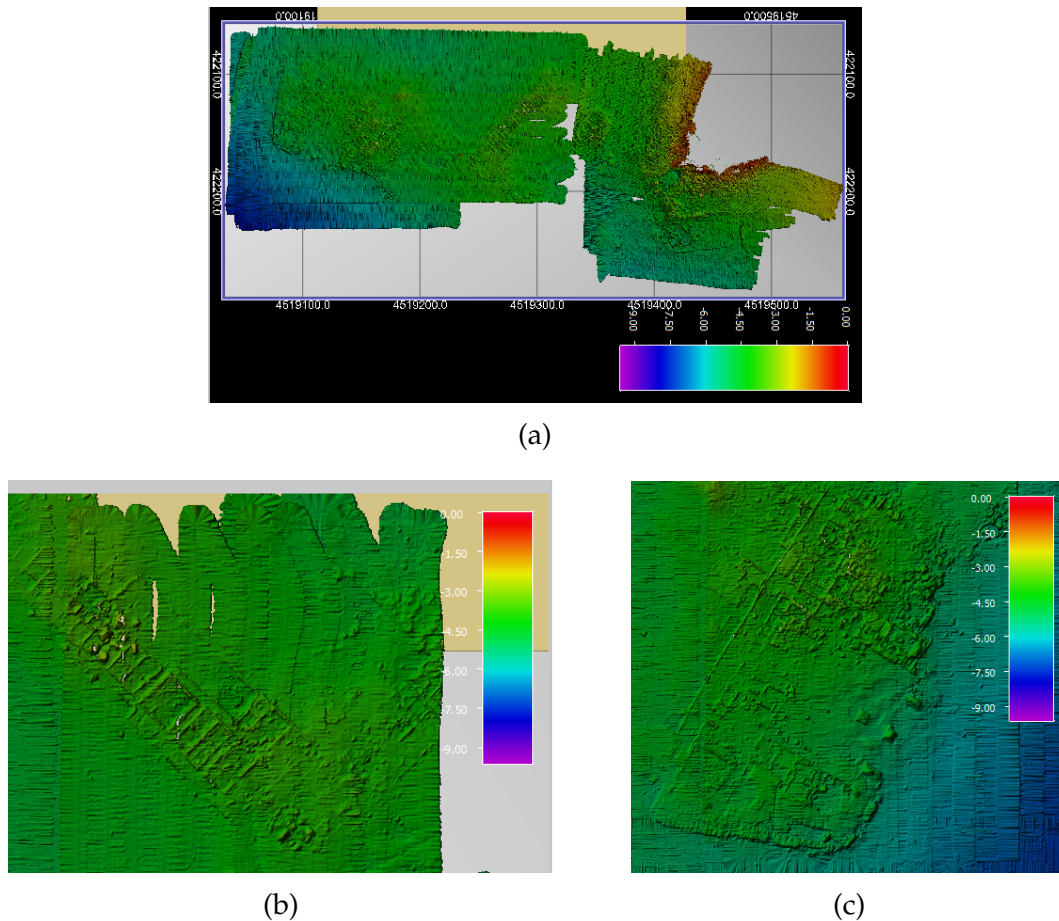
Figure 7.19: (a) Results of the whole Baia pilot site's bathymetry. (b), Details of the submerged Villa dei Pisoni. (c) Details of the submerged Villa Protiro. 2.5 bathymetric models are based on MBES data collected by the ASV PlaDyBath.

Results

The georeferenced side-scan sonar mosaic shown of the Baiae underwater archaeological park is shown in Fig.7.18. Results of multibeam sonar data postprocessing in QPS Qimera software are shown in Fig.7.19. Bathymetric map of the whole survey Baiae area is given in Fig.7.19a, with details of the sunken Vila dei Pisoni in Fig.7.19b, and details of the sunken Vila Protiro in Fig.7.19c. It is notable that the walls of these villas can clearly be distinguished from the surrounding seafloor.

Discussion

The precision of position in bathymetric model of Baiae site is on the order of $10cm$ because there we were not given access to the local base stations for georeferencing corrections through NTrip client. Since the beam of the MBES is around $1°$ this means that its minimum trace on the seafloor is $0.01h$ at the depth h. This means that for the depths of ancient Roman Baiae site in the range $6 - 8m$ the best resolution of sonar
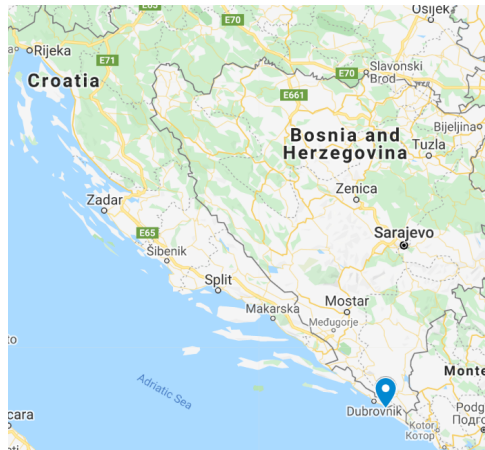
Figure 7.20: The location of the Cavtat underwater archaeological site

data was $6 - 8cm$. The interpolation grid size in QPS Qimera was $10 \times 10cm$ since higher resolution was not required by the archaeologists, and all the walls and masonry details of villas are visible in bathymetric maps. In the initial bathymetric interpolated model there were waves which translated from the surface to the sonar data. This is a consequence of the INS system not having a heave sensor. However, Qimera's Wobble Analysis tool and further spline interpolation solved this problem in the post-processing phase.

7.5.4 Cavtat sites results

The Adriatic Sea is teeming with undiscovered ancient shipwrecks and untold treasures. Two remarkable underwater sites in Croatia, one with around 700 amphorae and the other one with dolii are located in front of Cavtat, a town south of Dubrovnik, as shown in Fig.7.20.

Merchant - ancient Greek sailing boat with dolii

One special feature, though, is the underwater site officially discovered in 1996, representing the only preserved site of large ancient ceramic vessels for cargo - dolii (Greek: Pithos, Latin: dolium) on the eastern Adriatic coast. It is an intact location at the depth of around $20m$. The site covers $10 \times 20m$ area. Traces of the sunken boat have not been noted though the dolii's discovery may simply illustrate the unproven shipwreck. Throughout the 2nd and 1st centuries B.C., dolii are also found on Roman warships in the Antiquity. These became the primary way of shipping goods in the 1st century, and the features of ancient tankers were taken over by vessels that transported them. Their capacities ranged between 1,500 and 3,000 liters. Cavtat's dolii date back to the 1st century and its estimated capacity ranged from 1,200 to 1,400 liters.

Amphorae cage

It is an ancient Greek shipwreck believed to hold a cargo worth around $5 - 8$ million in today's value has been recently protected with a large cage to shield the ship's cargo from scuba looters who would be tempted to steal the objects. The wreck of these 700 vases was a Greek trade vessel of the second century bearing a shipment of earthenware amphorae of olive oil and wine, which sunk just off the coast of a small town of Cavtat, 20 km south of Dubrovnik, Croatia. In 1999, Boris Obradovic, the director of a nearby Scuba Diving center discovered the wreck and now guides more seasoned scuba divers down to the debris. The wooden ship is almost entirely decomposed, but oddly enough the ceramic amphora's holding the wine and olive oil are still intact and lined in the ships holds row after row. This freight that has considerable historical importance and interest on the black market is worth quite a bit of money, prompting the Croatian authorities to cover the wreck with a huge heavy duty metal frame. The cage itself is about 20 m long and 10 m high, and has a big hinged door that can be locked closed.

Methodology

In 2018, LABUST team went to Cavtat to record the dolii and amphorae cage sites which are only a few hundred meters apart. This was done under Interreg Mediterranean co-funded project BLUEMED. The team worked from a catamaran work boat anchored close to the underwater archaeological sites. The deployment of the bulky ASV was simplified by using a mobile manual crane operated from the spacious flat stern of the catamaran work boat. This enabled for the multibeam sonar mounted ASV to perform survey missions and record bathymetric data more easily, thus losing less time during the battery changes.

BLUEROV2 was used for visual data collection, as shown in Fig.7.23a, mainly at the doliaisite. Easily deployable, and manually controlled, ROV has been shown to be very useful, especially since there is a direct visual link through the tether back to the operator's screen.

One of the goals of this Cavtat trial was to gather dataset for photogrammetry of the Supetar islet close to these two underwater archaeological sites. The UAV took HD photos of the island in a crosshatch coverage plan, with 70% across- and along-track overlap between adjacent photos, at an altitude of 40m for safety reasons.

(a)

(b)
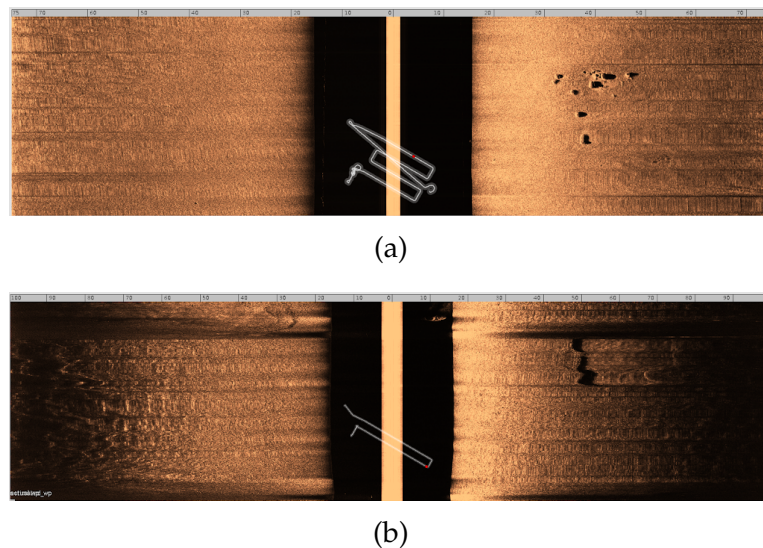
Figure 7.21: Side-scan sonar imagery of the sites. (a) Dolii site. (b) Amphorae cage site

Results

Precise positions of of both sites were determined from the georeferenced side-scan sonar imagery of the area, recorded by the LAUV Lupis, as shown in Fig.7.21. The resulting bathymetric map of the area is given in Fig.7.22.

In the post-processing phase, a 3D model of one dolium was generated from downsampling a 25fps HD video from ROV's logs, as shown in Fig.7.23b. The biggest problem with the use of frames extracted from the recorded video is motion blur, which renders 3D models of relatively less quality compared to the ones generated from still photographs of a DSLR camera with an external high-power flash. Nonetheless, this opens a possibility to document an underwater archaeological site much faster and with much less logistics compared to the case when divers record the site. The result of the photogrammetry post-processing of the Supetar islet aerial photos is given in Fig.7.23c.

Discussion

The precision of the navigation data at Cavtat site was on the order of $1cm$ since corrections w.r.t. the local CROPOS base station system were available through NTrip client connected over a 4G modem onboard the ASV. Since the beam of the MBES is around $1°$ this means that its minimum trace on the seafloor is $0.01h$ at the depth h. This means that for the depths of the amphorae cage and dolii site in the range $20 - 25m$ the best resolution of sonar data was $0.2m$. The interpolation grid size in QPS Qimera was $20 \times 20cm$ since higher resolution was not required by the archaeologists.
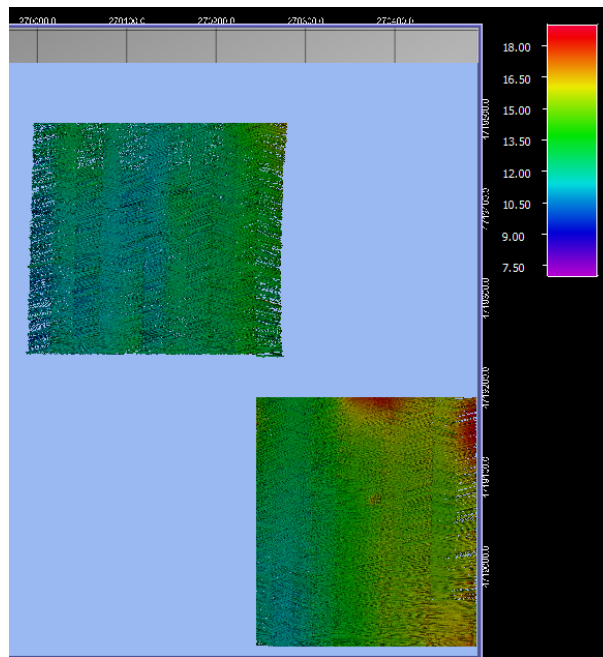
Figure 7.22: Bathymetry map of the underwater archeological sites in front of Cavtat, Croatia. 2.5 bathymetric model is based on MBES data collected by the ASV PlaDyBath.
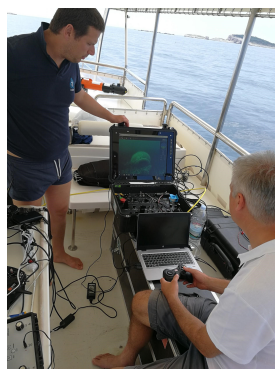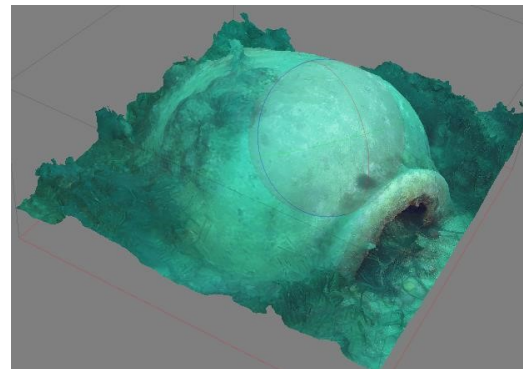
(a)   (b)

(c)

Figure 7.23: (a) Operating BLUEROV2 at Dolia site in Cavtat, Croatia, with a direct HD video feedback on a high-contrast screen.(b) Photogrammetric 3D reconstruction of a dolium based on frames from ROV's camera. (c) 3D reconstruction of the island Supetar in front of Cavtat bay, close to the diving locations of the amphorae cage and dolia locations.
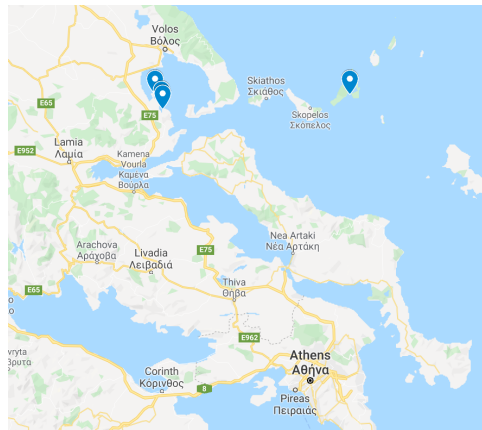
Figure 7.24: The locations of the underwater archaeological sites in Western Pagaseticos.

7.5.5 Western Pagaseticos sites

Four underwater archaeological sites were surveyed by our autonomous vehicles in the Westerns Pagaseticos, Greece, a map of which is shown in Fig.7.24.

Alonissos - Peristera shipwreck

This shipwreck was found early in the nineties. The uncovered pieces of the wooden ruin of Peristera have been rotting away for a long time, but the remaining cargo has a fantastic 4,000 amphorae seascape. The ship, claimed by historians to be a huge Athenian barge holding wine-filled amphorae, presumably sunk at the end of the fifth century B.C. The hull still lies submerged at around $30m$, and archaeologists believe it is the biggest ship of its type found beneath the sea.

At this site the team first obtained a georeferenced mosaic of the collected side-scan sonar data collected by the AUV Lupis around the pilot site is given in Fig.7.25a. The ASV was used to gather bathymetric data of the pilot site in more detail, with a narrow angle of view of the sonar, and high percentage of swath overlap between neighboring survey lawnmower lanes. A wider sonar angle of view, but again with high percentage of overlap between consecutive survey lanes were used to gather bathymetric data of the pilot site's surrounding area. The overlay of all survey missions planned as well as wide area detailed bathymetry map of the pilot site itself are given in Fig.7.25b.

UAV DJI Phantom 4 UAV was used to gather photos of the Peristera island part just in front of the pilot site. It was programmed to execute a crosshatch mission at altitude of 40m, and a 70% along- and across-track overlap between photos. Results of the 3D reconstruction of the Peristera island part is given in the Fig.7.25c, based on 381 photos.

Merging of the low-resolution acoustic model of the seafloor with the high-resolution textured photogrammetry model is shown in Fig.7.26, which is one of the results stem-

(a)   (b)

(c)
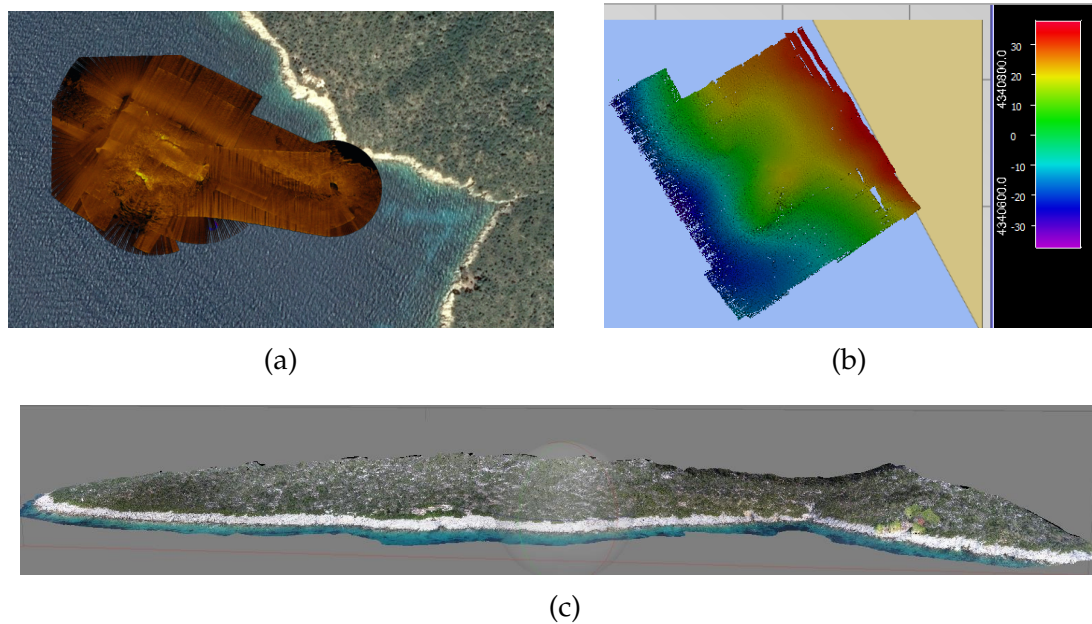
Figure 7.25: (a) Side-scan sonar mosaic of the Peristera site georeferenced and shown in Google Earth, based on SSS data collected by LAUV Lupis. (b) Results of bathymetry of the area surrounding the Peristera pilot based on MBES data collected by ASV PlaDyBath. (c) 3D reconstruction of the part of Peristera island in front of the pilot site locations, based on visual data collected by UAV DJI Phantom 4.

ming from the aforementioned BLUEMED project and collaboration with our partners from University of Calabria led by Fabio Bruno [107,148]. The reason why this merger was performed is that one of the BLUEMED project goals was to preserve the underwater cultural heritage sites in a digital form, so that visitors of the museum in the vicinity of the site can experience the diving visit to the site by wearing a virtual reality set. This allowed for the low-resulotion acoustic model of the area to provide the georeferenced collocation to the high-resolution textured photogrammetry model of the site, but also the general features for the UCH's surrounding area.

Kikinthos shipwreck - West Pagasetic gulf

Kikinthos islet is a small breakwater located east of the Bay of Amaliapolis. In ancient times, Kikinthos was used as a quarantine for the seamen who had returned from their voyages to Amaliapolis. The shipwreck was discovered under the auspices of the Hellenic Institute of Marine Archaeology and the guidance of the maritime archeologist Elias Spondylis during an underwater survey undertaken in 2005 at the northwestern end of Kikinthos Islet. The remains of a primarily pithoi Byzantine shipwreck cargo (large shipping containers) are found about $3 - 11m$ from the shore. Large pithoi fragments, which can be traced to at least three separate styles, occupy an area of about $8 \times 6m$. There are also fragments of two types of amphorae, dated from the 12th-13th

(a)

(b)

(c)
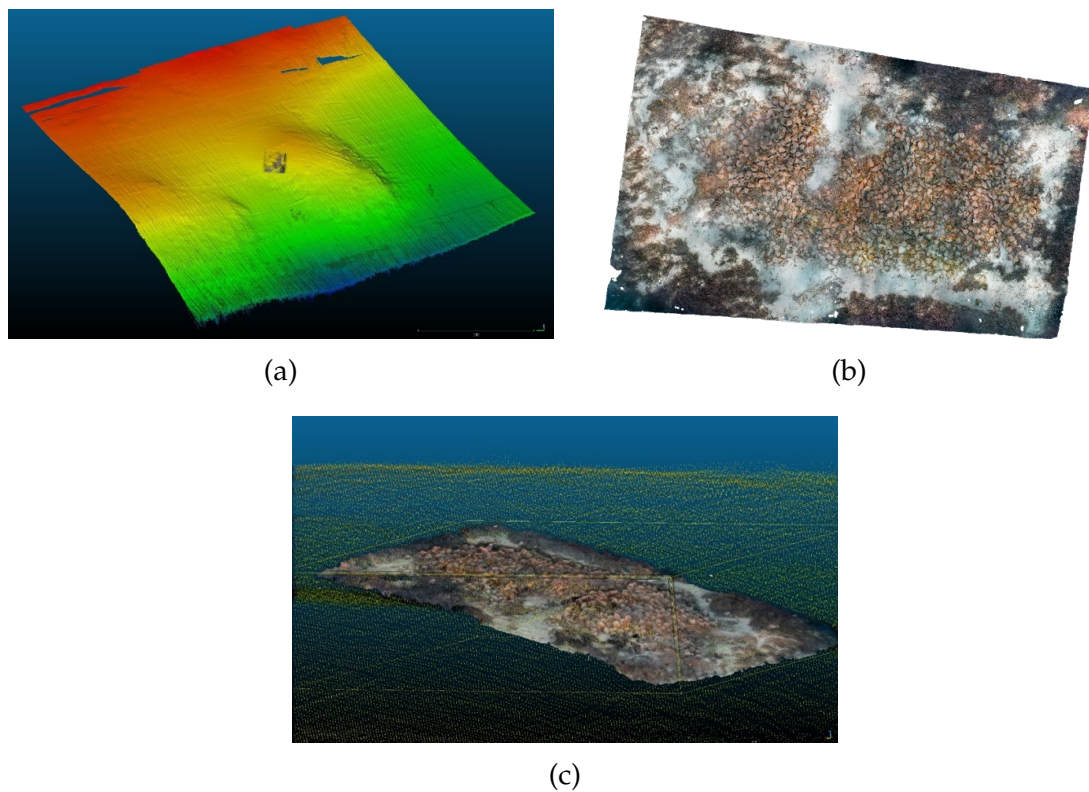
Figure 7.26: (a) Bathymetry of the area of the site based on MBES data collected by ASV Pla-DyBath. Small box in the middle is the collocated 3D photogrammetric model of the Peristera shipwreck site. (b) Orthophoto projection of the photogrammetric 3D model of the site based on high resolution photos taken by divers. Courtesy of Fabio Bruno. (c) Merge of the acoustic and optical 3D models [107,148]. Courtesy of Fabio Bruno.
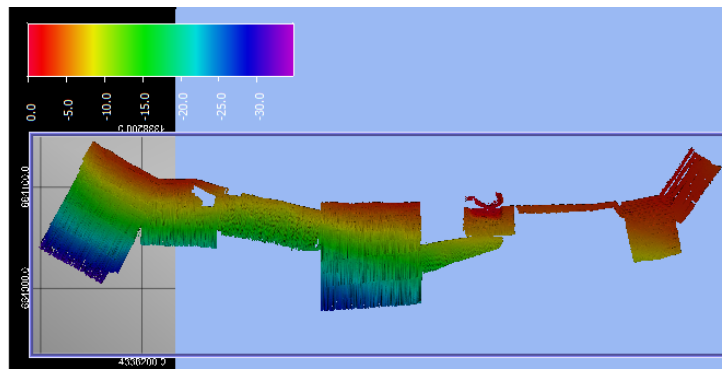
(a)
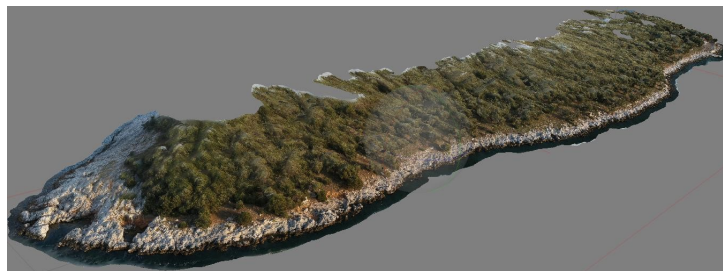
(b)

(c)

Figure 7.27: (a) Results of bathymetry of the area surrounding the Kikinthos site based on MBES data collected by ASV PlaDyBath. (b) and (c) 3D reconstruction of the Kikinthos island in front of the pilot site locations, facing towards Amaliapolis and the outer islet side, respectively. Based on images taken by UAV DJI Phantom 4.

centuries AD, in the pithoi. The styles of pithoi are traced to the 8th – 9th centuries AD, but it seems they coexist with the later amphorae as storage vessels were typically used for lengthy periods of time.

Bathymetry of the shipwreck and the whole inner side of the Kikinthos islet facing towards Amaliapolis is shown in Fig.7.27a. DJI Phantom 4 UAV was used to gather photos of the site's surroundings. On these pilot sites it was flown slowly in manual mode at an altitude of approx. $10 - 15m$ to ensure high percentage of photos overlapping. The camera was oriented directly towards the shore, to get more details of the shore. Photogrammetric 3D reconstruction of the whole Kikinthos island is shown in two parts: side facing towards Amaliapolis (see Fig.7.27c, and the other side facing out (see Fig.7.27c.

<center>(a)</center> <center>(b)</center>
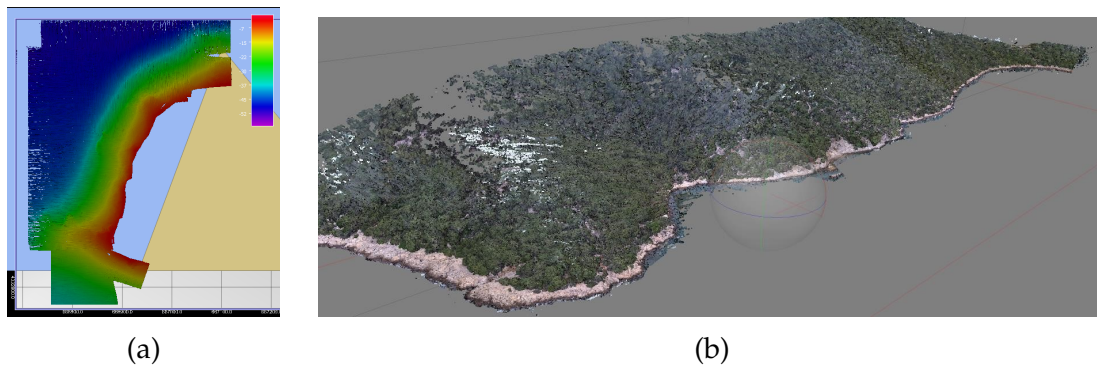
Figure 7.28: (a) Results of bathymetry of the area surrounding the Glaros site based on MBES data collected by ASV PlaDyBath. (b) 3D reconstruction of the Glaros cape in front of the pilot site locations. Based on images taken by UAV DJI Phantom 4.

Akra - Glaros shipwreck

This archaeological underwater site is located in an area opposite of Nies, a coastal village in Magnesia Prefecture and near Amaliapolis City. According to the Hellenic Institute of Marine Archaeology, which investigates the area from 2000 to the present under the direction of the marine archeologist Elias Spondylis, at least four shipwrecks were recognized: the Hellenistic one (3rd – 2nd century BC), the Early Roman one (1st – 2nd century AD) and two of the Middle and Late Byzantines (12th – 13th century AD) where late Roman pottery is also present. The reports related to the above shipwrecks are so scattered and confused that the classification of the different shipwrecks is an activity that is very challenging and not yet complete.

Bathymetry of the shipwreck and wide area around the underwater archaeological site is shown in Fig.7.28a. Photogrammetric 3D reconstruction of the whole Glaros cape shore is shown in Fig.7.28b.

Telegrafos shipwreck

Telegrafos Bay is situated in Magnesia Prefecture and near the town of Amaliapolis. The shipwreck was first discovered in 2000 by the Hellenic Institute of Marine Archaeology team that explored the region of the south-western coast of the Pagasetic Gulf, and was then thoroughly excavated from 2003 to 2008, also by HIMA, under the supervision of the marine archaeologist Elias Spondylis. Unfortunately, previous to the excavation the site was robbed. Nonetheless, the discovery brought the majority of the cargo to light, and a detailed analysis contributed to the identification of three major forms of Late Roman (4th century AD) amphorae for the major cargo which could be traced to northern Peloponnesian (Corinth) and Eastern Aegean (Samos) roots. Facts show a ship sailing down the Late Roman sea routes-the recently formed Byzantine
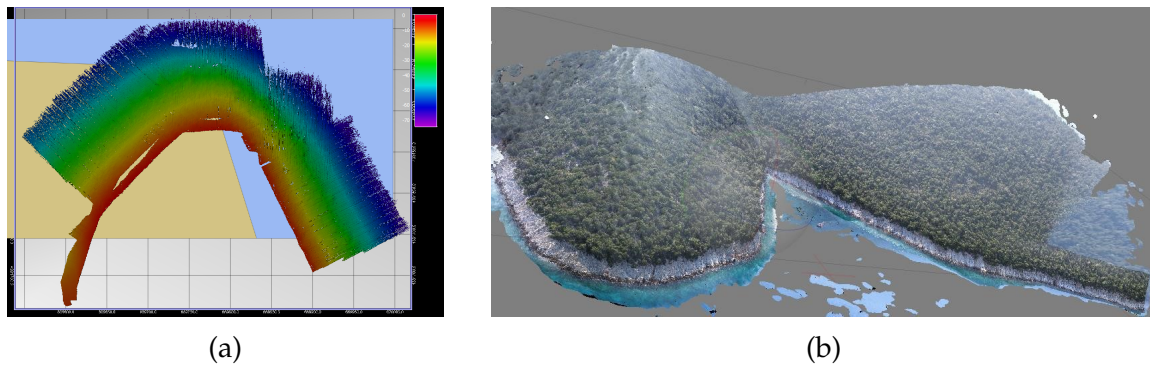
Figure 7.29: (a) Results of bathymetry of the area surrounding the Telegrafos site. (b) 3D reconstruction of the Glaros cape in front of the pilot site locations.

Empire through the Aegean Sea and down its edges.

Bathymetry of the shipwreck and wide area around the underwater archaeological site is shown in Fig.7.29a. Photogrammetric 3D reconstruction of the whole Telegraphos cape shore is shown in Fig.7.29b.

Discussion

The precision of the navigation data at all four Wester Pagaseticos UCH sites was on the order of $10cm$ because corrections from local base stations were not available to us. Since the beam of the MBES is around $1°$ this means that its minimum trace on the seafloor is $0.01h$ at the depth h. At Peristera, Glaros and Telegrafos UCH sites the measured depth was on a range $0 - 60m$. The Peristera shipwreck is located in the area where the depth is $20m$. For these shallow parts of the surveyed area a $20 \times 20cm$ interpolation grid size in QPS Qimera was used. For deeper parts of the survey area grid cell was set to $30cm$ since they were needed only for general seafloor morphology when merging opto-acoustical model of the shipwreck in virtual reality. The same applied to Glaros and Telegrafos UCH sites since they are the remaining artifacts after shipwrecks lays no deeper than $20m$. At Kikinthos UCH site the measured depth was on a range $0 - 30m$. Here interpolation grid cell size of $20 \times 20cm$ was used for the whole surveyed area.

7.5.6 Szent Istvan shipwreck

S.M.S. Szent Istvan, the only ship belonging to the Hungarian monarchy, met her end on June 10th 1918 shortly before dawn. It was sunk by Italian torpedo boats. On the 101st anniversary of this event the shipwreck was recorded for the first time by our multibeam sonar-mounted autonomous surface vehicle. The shipwreck has already suffered irreversible degradation of her steel and iron hull. Thus, the main objective
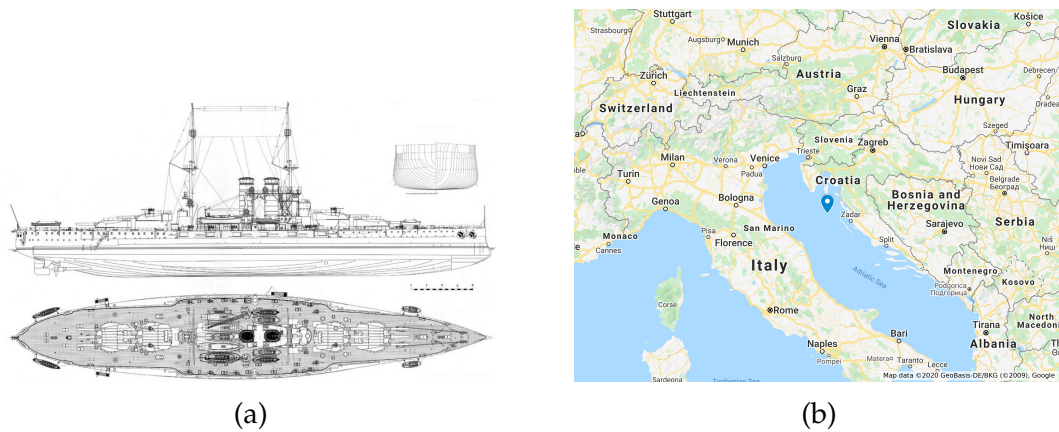
Figure 7.30: (a) Blueprint of the cross section, side and top view of the SMS Szent Istvan, [152]. (b) Location of the UCH site SMS Szent Istvan shipwreck.

of the bathymetric surveys was to assess the current state of the shipwreck and to set up a foundation which future monitoring operations could be built upon and compared with. SMS Szent István was an Austro-Hungarian battleship of the Tegetthoff class, constructed in Rijeka and Pula and completed in 1914. It was the only Austro-Hungarian ship to serve the Hungarian part of the monarchy, [152]. Its blueprints are given in Fig.7.30afor reference in the remainder of this subsection.

The Szent Istvan wreck (the deepest point at 68 meters) has been visited so far by many local and foreign divers. It lies inverted with the deck facing the bottom, with the cannons still facing left. Drawings of the shipwreck by Danijel Frka are given in Fig.7.31. In Fig.7.31adetails of the southern side of the ship's aft are shown, namely propellers, motor shafts, cannons, keel, as well as numerous fishing nets laying on the shipwreck and around it. A depression can be seen just above the cannons that was the consequence of ship's hull imploding the air trapped inside during the sinking. Moreover, from the shadow below the southern side of shipwreck's aft it can be deduced that the shipwreck is leaning a bit to its northern side, thus creating an opening under the ship's hull on its southern side, [152]. Details of the northern side of the torn bow, which broke during the sinking, are drawn in Fig.7.31b.

Methodology

As shown in Fig.7.32a, the shipwreck lays upside down on the floor, leaning on its superstructure and resulting with the south side of the hull been lifted from the seabed. The size of the gap on the stern part of the ship is shown in Fig.7.32b. Degradation of the steel hull underwater would eventually result in closing that gap until the hull completely collapses under its own weight. To measure the gap along the ship side, the profiling sonar carried by autonomous surface vehicle was used utilizing the method-
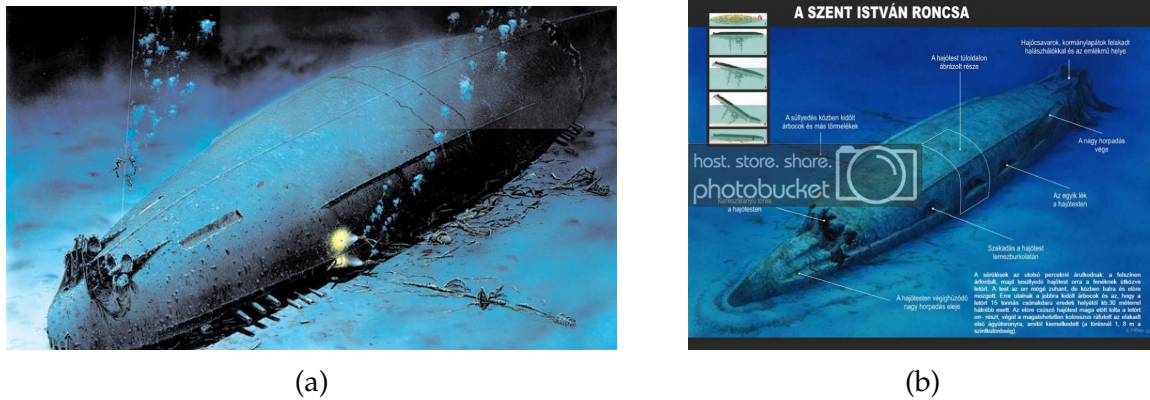
Figure 7.31: Drawing of the (a) aft (b) bow of the SMS Szent Istvan shipwreck. Image courtesy of Danijel Frka, [152].
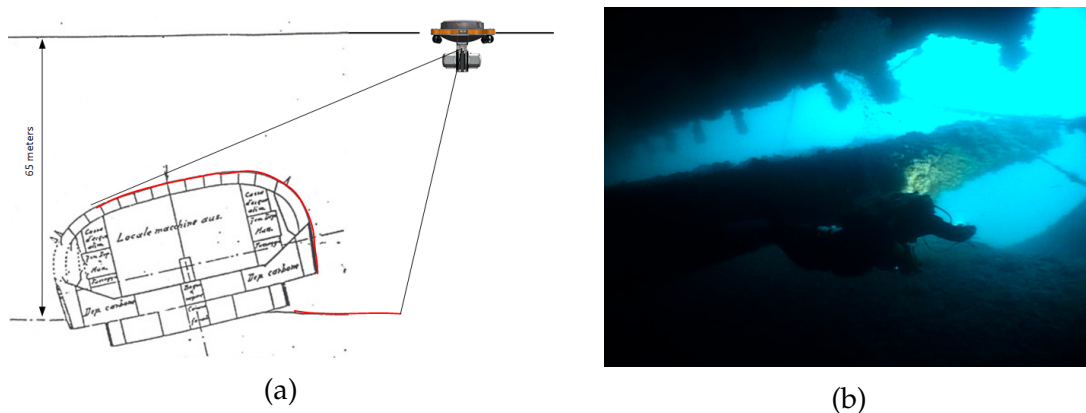
Figure 7.32: (a) Methodology for measuring the Gap using ASV and the profiling multibeam technology. Red line represents the sonified area. The image is conceptual, it does not respect the real proportions e.g. depth vs. hull size, nor sonar beam width and tilt angles. (b) Image representing the size of the gap between the ship hull and the seabed relative to diver. Image courtesy of Marino Brzac.

ology shown in Fig.7.32a. It was necessary to design a mission for the ASV PlaDyBath, which runs 50m from the south and north sides of the ship and parallel to the ship, as shown in Fig.7.33by yelow lines. The viewing angle of multibeam sonar was set to 60°, but the rays tilted to the left by 15°, as shown conceptually in Fig.7.32a.

Bathymetry survey results

In total, an area of 200x75m around the site was recorded by multibeam sonar, using standard lawnmower missions along, across, and from the sides of the wreck, as shown in Fig.7.33by white lines. In missions planned along and across the wreck to capture as much detail as possible, the sonar viewing angle was also set to 60°, but without ray tilting, and with adaptive ping frequency to ultimately obtain the highest quality shipwreck model from the sonar data. The navigational precision of the autonomous vehicle, and therefore the precision of geolocation of the 3D model reconstructed from
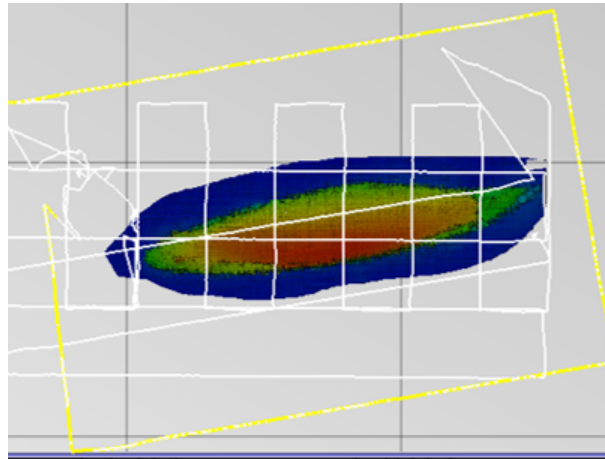
Figure 7.33: Overlay of ASV PlaDyBath's survey paths w.r.t. the top view of the bathymetric model of the shipwreck. (yellow) survey missions around the ship with tilted sonar beams to record the side of the ship (white) standard lawnmower missions along and across the wreck to capture the general morphology of the shipwreck.

sonar data, is of the order of 10 cm, which is more than sufficient for archaeological applications.

The missions planned in advance for the surface vehicle were in the form of transects spaced 25m apart, thus providing sonar data with much redundancy and more detail in an important part of the shipwreck area. QPS Qimera software was used to reconstruct the bathymetric model from sonar and navigation data. A 0.5m resolution was used for the bathymetric model. The wreck length measured from the reconstructed 3D model created from the multibeam sonar data was 145m, the wreck width was 28m, and the bearing direction from the stern to the bow (bearing angle) was 79.4°. Size of the 3D model matches the real size of the ship very accurately. Precise coordinates of the stern and bow centers in the WGS84 system were also obtained.

Fig.7.34a shows the reconstructed bathymetry model of S.M.S. Szent Istvan shipwreck seen from its northern side. It is interesting to note how the sides of the shipwreck are very steep, almost vertical. This could be the consequence of a high number of outliers in the point cloud in these areas due to the fishing nets hanging all over the shipwreck. The ship's propellers, motor shafts and the depression on its aft side are clearly visible in the model, as shown in Fig.7.34b, which shows the aft of the ship from its southern side with a great similarity to the drawings of the shipwreck in Fig. 7.31a. Also, the torn bow part of the ship is shown in Fig.7.34c, which is also identical to the drawings of Danijel Frka given in Fig.7.31b.

Another interesting finding and confirmation of the reports we received from the Szent Istvan wreck divers is the opening below the south side of the ship. To record this with the multibeam sonar, its beams had to be tilted in order to catch the morphology of the opening.

(a)

(b)

(c)
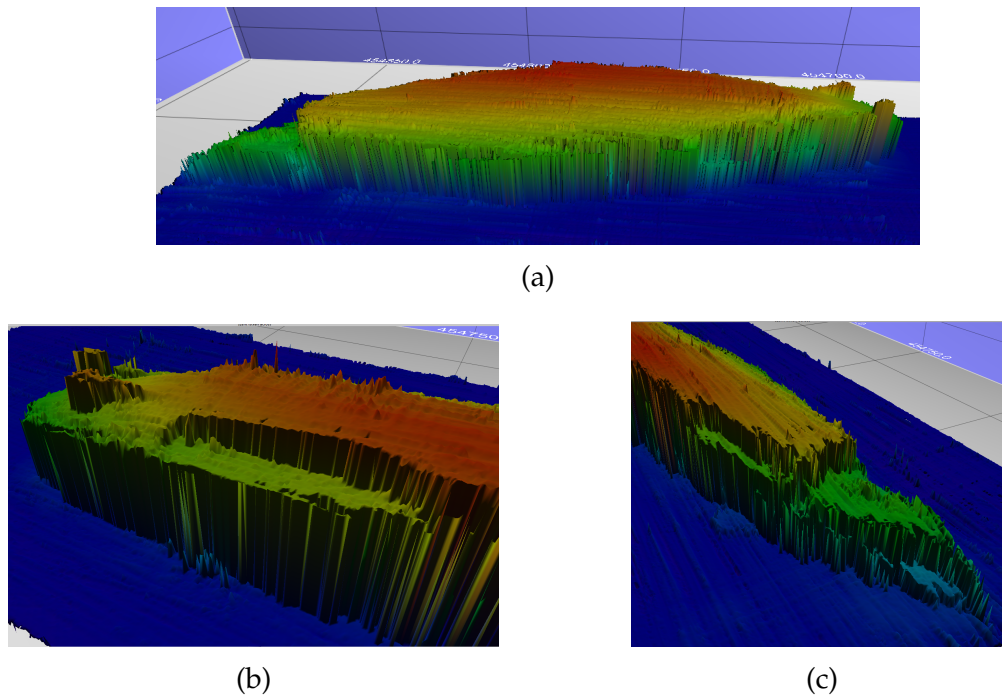
Figure 7.34: (a) Bathymetry model of S.M.S. Szent Istvan seen from its northern side. (b) Bathymetry model details of the aft seen from its southern side: the depression in ship's hull, propellers, and motor shafts are clearly visible. (c) Bathymetry model details of the torn bow seen from its southern side. Based on MBES data collected by ASV PlaDyBath.

This opening could not be faithfully reconstructed in 3D using Qimera since it only interpolates a 2.5D manifold. Instead, characteristic transverse profiles of the south side of the stern of the ship clearly show an opening 4m high, extending 100m along the south side of the ship and entering an average of 3-4m towards inside the ship, shown in Fig.7.35a. Further analysis of the ping point cloud shown in Fig.7.35a consisted of detecting and clustering points belonging to the seafloor and the shipwreck by thresholding depth values. The plot showing these two clustered point clouds is given in Fig.7.35b. The opening between these two structures is clearly visible as in some previous figures. However, this way the size of the opening can be assessed numerically, and the degradation level of the shipwreck's metal hull can be numerically represented through time with further monitoring missions.

Side-scan sonar survey results

The shipwreck and the surrounding area were also recorded by side-scan sonar installed on the LAUV Lupis, again using lawnmower missions along and across the shipwreck. The missions were planned first in relation to the buoy located on the stern of the wreck, and after processing the bathymetric data, the missions were designed in relation to the exact coordinates of the bow and stern. Since the weather conditions on the surface of the sea do not affect the operation of the underwater vehicle, it was used
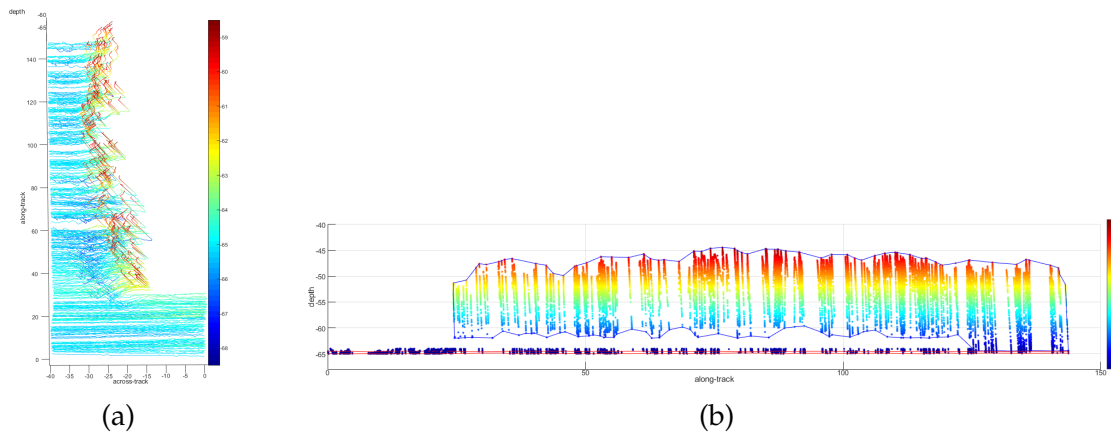
Figure 7.35: Acoustic returns of the multibeam sonar: (a) 3D line plot of the pings as a spatial representation of the opening below the southern side of the shipwreck. (b) Clustered and separated point clouds of the seabed and the shipwreck with the opening clearly visible in between. View is from the southern side of the shipwreck, so aft is on the left.

for filming the second day of the expedition on June 11, 2019. when the wind began to rise and create significantly larger waves than on the first day of the expedition.

After the first couple of missions, it was noticed that the navigation accuracy of the AUV Lupis, despite the presence of a DVL sensor (which is usually used to compensate for deviations from the given trajectory due to sea currents), accumulated an error over time. Since the Szent Istvan wreck contains a huge amount of metal, it caused the compass in the vehicle to deflect, thus negatively affecting the accuracy of the localization. To address this problem, subsequent missions are planned to compensate for vehicle trajectory drift due to external interference (course errors), and based on experience from previous missions.

A composite image of side-scans port and starboard side is given in Fig.7.36a. The upper left side of the image shows the torn aft; many fishing nets hanging from the ship's side can be seen on the upper right side of the image; the structure of plates along the ships hull can be seen from the left part of the image; keel is visible in the right middle part; propellers and rudders can be clearly seen from the bottom right side of the image. The georeferenced mosaic of side-scan sonar imagery exported to Google Earth is shown in Fig.7.36b. It is marked with points of interest (POIs) P1-P8. These POIs were detected in all the side-scan sonar missions but from different angles. Some of them were known to the underwater archaeologists with experience at this site, but some of them were sent in a report for further analysis with positions relative to the ship and the estimated size of the detected objects.

(a) (b)
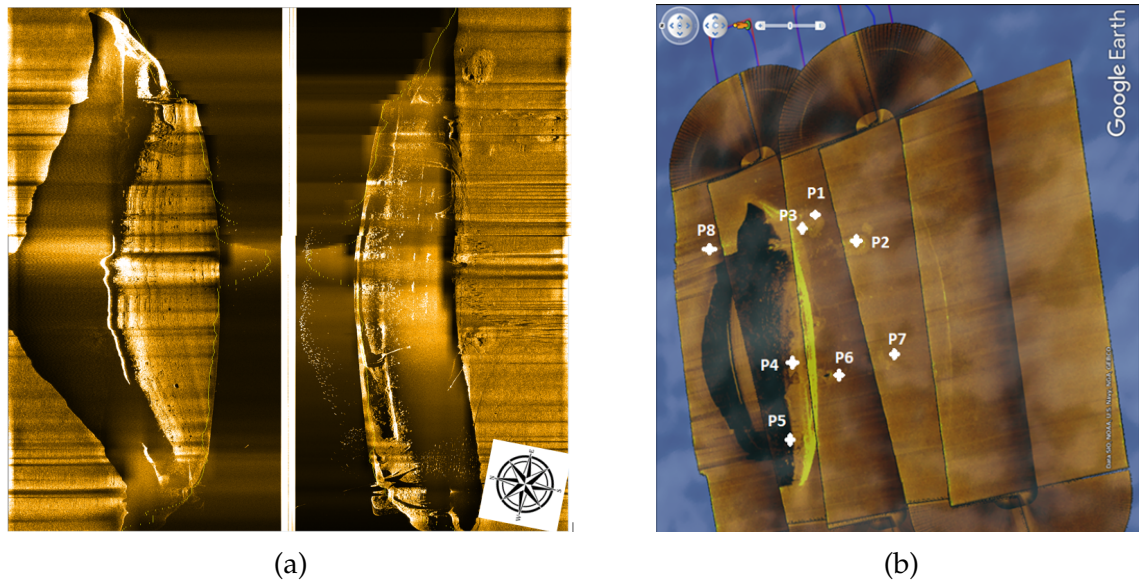
Figure 7.36: (a) Composite of side-scan sonar images of the shipwreck SMS Szent Istvan taken from the mission during which LAUV Lupis crossed directly above the wreck. (b) Geolocated mosaic of side-scan sonar data from one of the Lupis AUV missions exported in .kmz format and imported into Google Earth with marked points of interest.

AUV's visual inspection results

In addition to side-scan sonar, the missions that used AUV Lupis to record the wreck of SMS Szent Istvan and surroundings used an integrated CCTV 1.4MP camera for visual inspection of the wreck in missions where the vehicle is planned to move a few meters above the wreck. The mission of the AUV Lupis was planned to move at a constant depth and along the ship's line from the stern to the bow, which we determined precisely from the bathymetric 3D model. Below are a couple of examples of interesting things captured by AUV's camera. Since the AUV had an altitude of less than $10m$ when crossing over the wreck, its LED flash burned the middle of each frame quite a bit. Red and yellow hues were suppressed as expected, and the images are bluish, which was expected because of the depth at which the shipwreck lays.

An example of an image of the shipwrecks hull taken by AUV Lupis camera is shown in Fig.7.37. The original image is shown in Fig.7.37a. Fig.7.37bshows the original burnt image filtered by the Contrast-Limited Adaptive Histogram Equalization (CLAHE) [151] in LAB color space. Fig.7.37cshows the result of applying the Adaptive Histogram Equalization (AHE) algorithm [153] to YCrCb color space of the original image. Lastly, Fig.7.37dshows the result of applying the Adaptive Histogram Equalization (AHE) algorithm [153] to YUV color space of the original image. It is noticeable that CLAHE, as an adaptive algorithm for cropping and equalizing the histogram of the brightness of the image, gives the best results with the fewest artifacts either by brightness or color change.
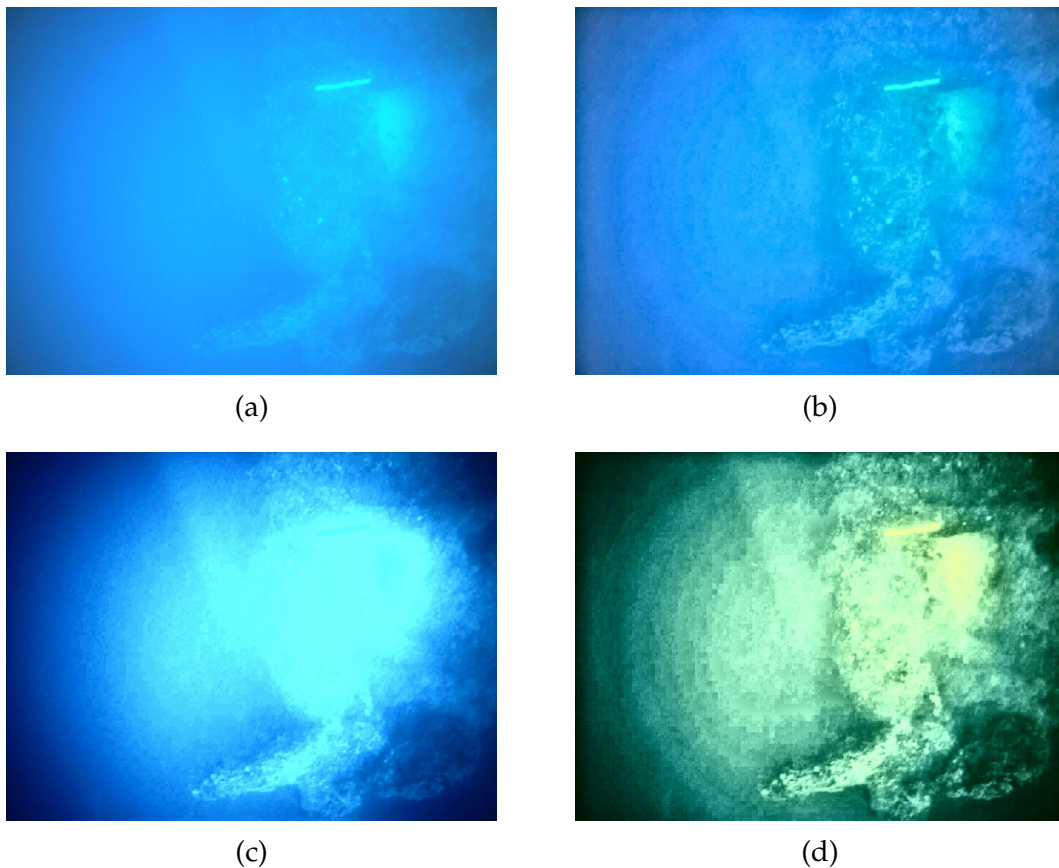
Figure 7.37: An example of brightness equalizing algorithms applied to visual data recorded by AUV Lupis. (a) Original image. (b) Image filtered by CLAHE algorithm in LAB color space. (c) Image filtered by HEQ algorithm in YCrCb color space. (d) mage filtered by HEQ algorithm in YUV color space.

Also, video frames captured by the AUV Lupis camera were used to generate 3D models and orthophoto projections (see Fig.7.38) in Agisoft Metashape software with High settings on Photo Alignment, Point Cloud, and Mesh. Area covered by this orthophoto is $4 \times 80m$. Although the 3D model is not georeferenced (though with additional processing of AUV Lupis logs, each frame can be paired with the estimated AUV Lupis position), nor does it represent the full picture of the bottom of the sunken ship Szent Istvan, it provides insight into how fast and easy autonomous underwater robots can be used. In a short period of time they record the area that divers would record for hours or days due to the limited duration of the dive at the depth of almost $70m$ at which the shipwreck lays.

7.5.7 Conclusion

A great potential for use of the autonomous marine vehicles in remote sensing surveys is presented in this section. Here three marine vehicles' and one unmanned aerial vehicle's use in nine underwater archaeology related autonomous remote sensing survey

(a)

(b)
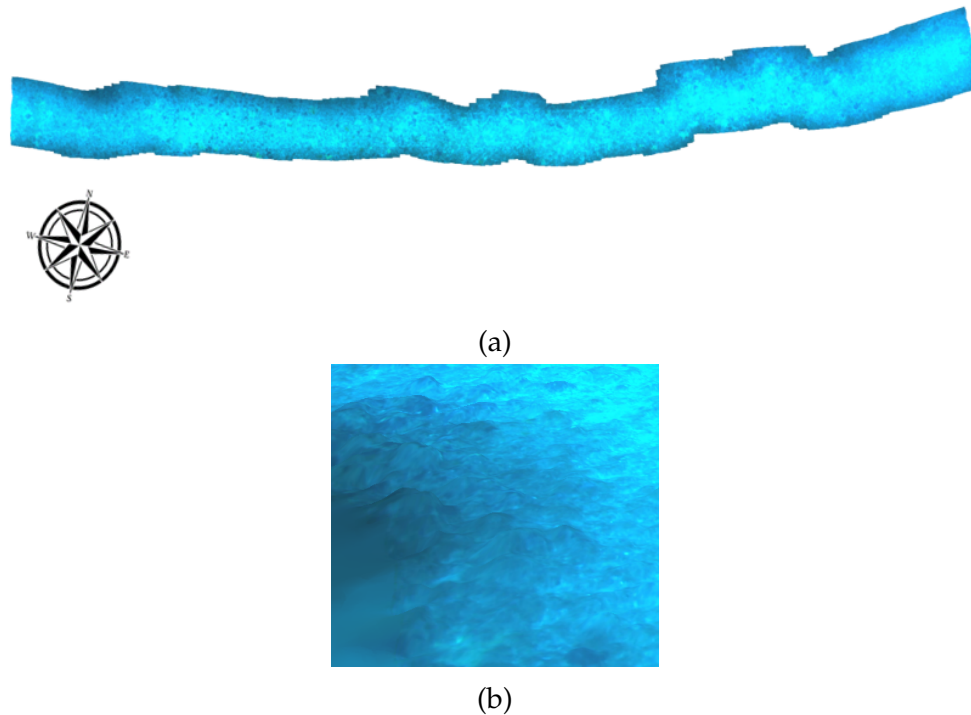
Figure 7.38: (**a**) Orthofoto mosaic of the top of the wreck obtained from the AUV Lupis camera images. Area covered: $4 \times 80m$. (**b**) A detail from the orthophoto showing level of detail of the orthophoto and biofouling on the Szent Istvan shipwreck hull.

missions is described through the surveying methodologies which yielded the presented results. The efficiency of using autonomous vehicles for such applications, data quality, quantity in unit time, as well as precision of georeferencing the data, especially on larger areas as in the authors' case studies, has been thus shown to far surpass the divers' performance.

The precision of bathymetry georeferencing ranged from $1cm$ when the local base station corrections were available to $10cm$ when only Applanix INS with Trimble GPS antennae were used. The resolution of bathymetric models ranges from $20cm$ for shallow areas where most of the UCH sites were located to $50cm$ for deeper parts. Position estimation of the AUV and SSS imagery was around $1m$ after smoothing the position estimation in the post-processing phase staring from the end to the beginning of the dive. The AUV has been shown to be at least $4 - 5$ times more efficient for visual inspection of the wrecks at depths over $50m$ compared to technical divers. Photogrammetric 3D models generated from photos taken by AUV, ROV and UAV show that even 20 minute autonomous survey missions can yield high-resolution models for various applications.

It is important to note that this was the first time that any of the chosen sites were documented by sonar technologies or autonomous marine vehicles. The main objective of the these surveys was to document and assess the current state of the sites and

to establish a foundation which future monitoring operations could be built upon and compared with. Also, going beyond mere documentation and physical preservation, examples of using these results for digital preservation of the sites in augmented and virtual reality are also presented.

7.6 Side-scan imagery processing

7.6.1 Real dataset results

In order to validate the results mentioned in Section 4.4, numerous real side-scan sonar datasets were acquired from field trials with the UUV in mostly underwater archeological sites near Cavtat, Croatia, Baiae Bay, Italy, Peristera island, Kikinthos island, Glaros and Tilegraphos Capes in Greece, as well as around Pelješac Penninsula and Biograd na moru in Croatia. The same five methods were then applied to the normalized side-scan sonar images from the real-world dataset. Parameters of all these methods and steps in the interesting object detection pipeline described in Section 4.3 are given in Table 7.3 with their values tuned to the real side-scan sonar data. Parameter $feature\_channels$ denotes what information from the image does a target detection method use, so value I represents image intensity/brightness, R represents contrast, and O represents orientation .

The resulting interesting objects' detection results are given in Figs. 7.39 and 7.40. The ground truth of what is interesting in the given sonar images was obtained from human operators circling the objects standing out from the usual clutter and noise in the side-scan sonar images. It can be noted that the anomaly detection method gives the most intuitive results in interesting objects' detection in the real side-scan sonar data as well, i.e. of all five methods it matches the human perception of "salient" and "outstanding" objects in the noisy side-scan sonar data the best.

7.6.2 Running on target hardware

After prototyping and performance testing phase, the anomaly detection method was implemented using OpenCV library and integrated into Robot Operating System (ROS) environment in order to benchmark its processing time on the target hardware onboard AUV Lupis, namely UDOO DUO. UDOO has a dual-core Advanced Reduced Instruction Set Computing Machine (ARM) $1GHz$ CPU, $1GB$ of RAM, and a Vivante GPUs for 2D, 3D and vector graphics. Specifications of UDOO DUO are given in Table 7.4:

Additional changes were made to the OpenCV implemented anomaly detection algorithm, namely converting the Nd convolution described in Section 4.3 into N $2D$

(a)

(b)
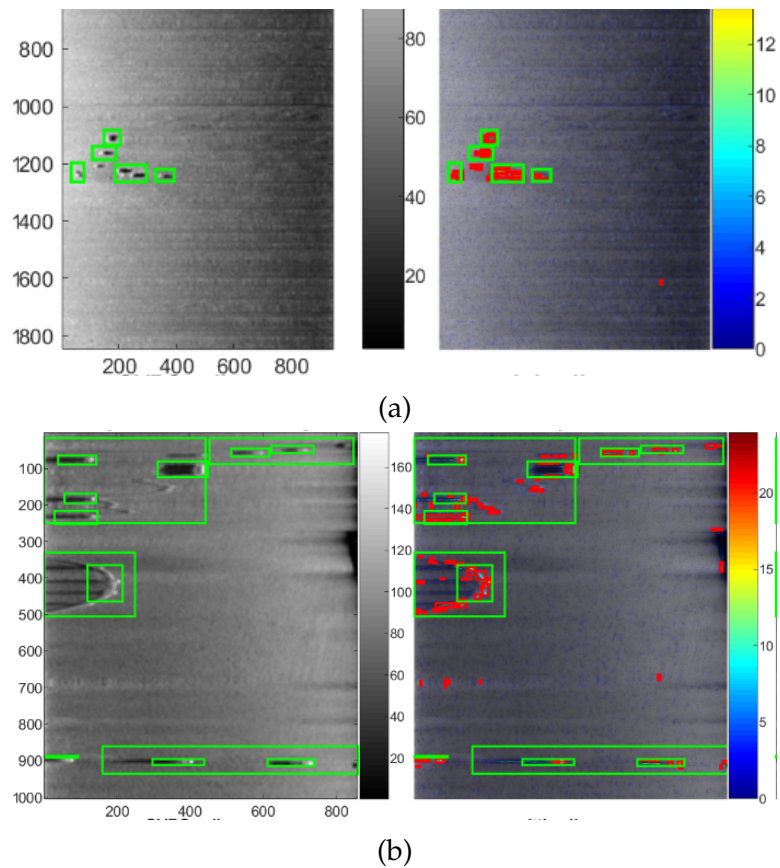
Figure 7.39: Examples of interesting objects detection in a real side-scan sonar image by various image segmentation methods. Colorbar represents the saliency metric normalized by its mean value to visualize which areas are the most interesting. Since anomaly and contrast methods practically detect edges in multiple scales, the anomaly/saliency values in the upper subfigures is covered with detection bounding boxes.

Table 7.3: Table of interesting object detection methods' parameters for real side-scan sonar images dataset.

	Anomaly Detection	Contrast-based Saliency	Itti-Koch Saliency	GBVS Saliency	Simpsal Saliency
mavg_M	10	10	10	10	10
gauss_kernel_w	10	10	10	10	10
gauss_sigma	1.6667	1.6667	1.6667	1.6667	1.6667
saliency_th	5	5	3	3	5
e_method	sobel	sobel	sobel	sobel	sobel
median_w	3	3	3	3	3
area_th	50x50cm	50x50cm	50x50cm	50x50cm	50x50cm
brigh_th	50.00%	50.00%	50.00%	50.00%	50.00%
scales	[1, 3, 5]	[1, 3, 5]			
blur_fraction			0.002	0.002	0.002
feature channels	IR	R	IO	IOR	IO

convolutions for computing local pixel histogram difference through the whole filter bank, since they are linearly separable and OpenCV did not offer a straightforward Nd convolution method. Also, Contrast-limited adaptive histogram equalization algorithm (CLAHE) algorithm, [151], was used to further normalize the brightness of the sonar images, thus improving the contrast towards slant range ends. Parameters of all these methods and steps in the interesting object detection pipeline described in Section 4.3 are given in Table 7.5 with their values tuned to the real side-scan sonar data.

Processing time for anomalous/salient object detection algorithm on UDOO is on average $50 - 60$ times slower than the processing time for anomaly detection on the computer. Still, UDOO manages to process 2-3MP side-scan sonar images in around $5s$ which is fast enough for mission replanning purposes, see Fig. 7.40a. It is also interesting to analyze processing time in a relative sense scaled by the number of MPs that the sonar image has. This is shown in Fig. 7.40b and can be used as a mission parameter. Number of across-track pixels in a side-scan sonar image for one straight line depends on sonar's across-track resolution and slant range, while it's along-track resolution depends on the surge speed of the vehicle, pining frequency of the sonar, and of course the length of the vehicle's path in one direction. Knowing the expected size of the sonar image per one path of the survey mission and the time needed to process

Table 7.4: UDOO DUO specifications.

dimensions	11 cm x 8.5 cm
CPU	2 x ARM® Cortex™-A9 core @ 1GHz with ARMv7A instruction set
GPU	Vivante GC 2000 for 3D + Vivante GC 355 for 2D (vector graphics)
RAM	DDR3 1GB
GPIO	76, fully available with Arduino compatible R3 1.0 pinout
Ports	HDMI and Low-voltage differential signaling (LVDS) + Touch 2 Micro Universal Serial Bus (USB) (1 OTG) 2 USB 2.0 type A 1 USB 2.0 internal pin header (requires adapter cable) Analog Audio and Mic jacks Camera Serial Interface (CSI) Camera Connection
boot device	Micro SD card
Power Supply	6-15V direct current (DC), and External Battery connector
Ethernet	Gigabit Ethernet RJ45 (10/100/1000 MBit)
WiFI	WiFi Module
External memory	Serial ATA (SATA) connector with power header

Table 7.5: Table of parameters for anomaly detection method implemented in OpenCv for real side-scan sonar images.

	Anomaly Detection
clahe_clip_limit	4
clahe_grid_size	32
gauss_kernel_w	10
gauss_sigma	1.6667
saliency_th	2.5
e_method	sobel
median_w	3
area_th	50x50cm
brigh_th	50.00%
scales	[1, 3, 5]

$1MP$, mission operator can decide to cut the sonar image of the whole line in smaller subregions in order to maintain the online sonar image processing and consequential mission (re)planning in case something interesting appears in the sonar data.

CPU usage on UDOO when anomaly-detector node enters the $getAnomalyMap$ method (see Section 4.3, Algorithm 3) is at most $90 - 100\%$ but only on one core of UDOO's CPU, leaving the other core unblocked for other operations. Memory usage is primarily affected by the depth n_{scales} of the filter bank, defined Section 4.3. In the tests, the average size of an image was $10MB$. Anomaly detector node used on average $(n_{scales} + 1) \times 10MB$. Since UDOO has $1GB$ of RAM and $1GB$ of swap memory, and effective filter depth for anomaly detection was empirically proven never to be $n_{scales} < 4$, this is than enough RAM for such applications.

7.6.3   Conclusion

The anomaly detection method from [95] had the best precision and recall performance on simulated sonar images dataset. Its performance was then validated on a real 500 side-scan sonar images dataset, and it also had satisfactory recall-precision performance. After prototyping and performance testing phase, the method was implemented using OpenCV library and integrated into Robot Operating System (ROS) environment in order to benchmark its processing time on the target hardware on-
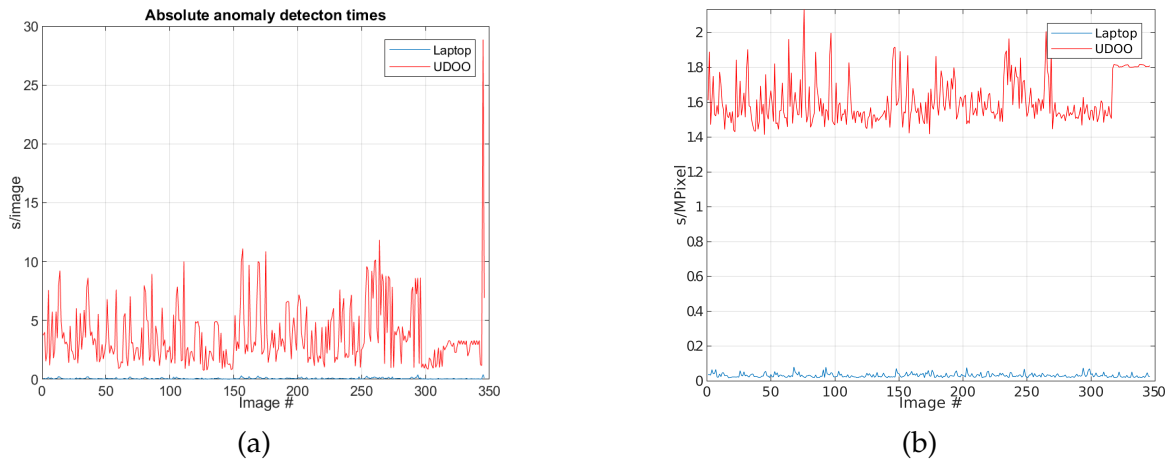
Figure 7.40: (a) Absolute processing time plot UDOO vs. laptop. (b) Unit processing time per megapixel UDOO vs. laptop.

board AUV Lupis. It turned out to be fast enough to process 1MP in $1.5 - 2s$ which is fast enough for large volumes of sonar data being recorded. Parts of this section were previously published by the author in [78].

7.7 Conclusion

This chapter presented experimental case studies of various subsystems, i.e. model predictive line following controller, extended Kalman-based underwater localization module, and hardware-in-the-loop side-scan sonar imagery processing module. It also described results of extensive trials with autonomous marine vehicles used in limnological and underwater archaeological applications. Autonomous operator experience gathered during these trials had a significant impact on the design of the proposed coverage path planning algorithms including many limitations of using side-scan sonar as a mapping sensor. It also resulted in a significant side-scan sonar imagery dataset later on used for performance validation of side-scan sonar processing algorithms.

Chapter 8

Conclusion

The goal of this thesis was to develop an online side-scan sonar data-driven coverage path planning (CPP) algorithm for monitoring and surveying large-scale (over $1km^2$) seafloor regions by an AUV, which would be much more convenient and less costly to deploy than a towfish. Furthermore, it is also important to scan parts of the coverage area in more detail and from more than one side if interesting objects are detected in sonar imagery. Interesting objects are assumed to be sparse in otherwise uniform sandy/muddy/stone seafloor. To accomplish that, four novel coverage path planning algorithms were developed. The proposed algorithms provide a coverage solution based on local information gain computed from the side-scan sonar data during the mission execution, which is then exploited for replanning. Theoretical analysis of the coverage planning algorithms is presented in a form of upper and lower performance improvement bounds w.r.t. the nonadaptive lawnmower coverage pattern. Extensive simulation tests in a grid-like coverage map were performed to get algorithms' performance statistics for a wide range of mission parameters. A few of the best performing coverage algorithms were then integrated into a 3D simulator to further validate the performance of the stated algorithms. Moreover, several modules of the NGC system and coverage policies were tested in experimental environment on a range of autonomous marine vehicles. The thesis stated several hypotheses and three major contributions in Section 1.3. These contributions are restated and reviewed in the context of the presented work.

The first contribution stated:

- *An online sonar data-based coverage path planning algorithm for large-scale seabed exploration missions using autonomous marine vehicles.*

In Chapter 3 four online sonar-data based coverage path planning algorithms are proposed. Section 3.5 presents a coverage algorithm based on propagation of sonar data information gain in a cost map. Three heuristic decision making-based coverage

algorithms rooted in lawnmower coverage pattern with online local replanning based on available sonar data are presented in Sections 3.6, 3.7, and 3.8. Algorithms presented in Section 3.7 and Section 3.8 are extensions of the heuristic coverage algorithm presented in Section 3.6. Theoretical upper and lower performance bounds of these algorithms are analyzed in Sections 3.5.6, 3.6.2, 3.7.2, and 3.8.2 w.r.t. the nonadaptive lawnmower coverage pattern. The analysis is based on a finite set of possible cases that can happen when the vehicle is moving forward/backward along the coverage area and detects something interesting in its left/right/both swath(s).

The second contribution stated:

- *Simulation framework for implementing, testing, and evaluating performance of sonar data-based coverage planning algorithms.*

All four proposed coverage planning algorithms were extensively tested in simulation. First, a simulation setup with a grid-based coverage map and varying mission parameters affecting the performance of these algorithms was made. It varied on a wide range the total area to cover, the ratio between the coverage area length and side-scan sonar swath, as well as the percentage of that area containing any interesting objects of random size and position. Setup and results of these simulations are presented in Sections 3.5.8, 3.6.4, 3.7.3, and 3.8.3. The results show a significant decrease of coverage path/mission duration w.r.t. the nonadaptable lawnmower pattern, especially in the intended range of mission parameters.

The best performing coverage algorithms were then integrated into the existing 3D ROS/Gazebo-based UUV simulator, as described in Chapter 5. This was done to gain further insights into the interactions between coverage planning, feasible path interpolation, control, as well as sonar image processing modules in a more realistic, physics-based simulation environment. Results of these simulations, presented in Section 5.4, validated the grid-based coverage map simulations.

Moreover, noisy side-scan sonar imagery was simulated to choose between several anomaly detection methods, as presented in Chapter 4. Most of these methods were previously used on natural images so for some of them certain adjustments needed to be done to apply them to sonar imagery.

Apart from validations in simulation, Chapter 7 brings five experimental case studies using various autonomous marine vehicles described in Chapter 6. These case studies are related to performance validation of subsystems such as model predictive line following guidance system (Section 7.2), underwater vehicle localization system based on sensor fusion (Section 7.3), sonar-based coverage policies in challenging nonuniformly shaped area such as lakes (Section 7.4), numerous underwater archaeological sites (Section 7.5), as well as hardware-in-the-loop performance analysis of sonar im-

agery anomaly detection methods (Section 7.6).

The third contribution stated:

- *Metrics for evaluating performance of sonar data-based coverage planning algorithms used in large-scale underwater survey missions.*

Behavior and performance of the proposed coverage algorithms was first analyzed on a unit coverage area to generalize algorithm behavior. Best- and worst-case performance w.r.t. the nonadaptive lawnmower coverage pattern are identified for all four presented algorithms, as presented in Sections 3.5.6, 3.6.2, 3.7.2, and 3.8.2. As mentioned above, extensive simulations were performed with a wide range of coverage mission parameters to validate the theoretical performance bounds of these algorithms. These analyses are given in Sections 3.5.8, 3.6.4, 3.7.3, and 3.8.3. Section 3.9 aggregates all these metrics, i.e. coverage path/mission duration decrease, chance of each adaptive coverage planning algorithm to have better performance as a function of mission parameters, limit percentage of coverage area containing interesting objects for which even adaptive behavior yields on average longer coverage paths, as well as total (re)planning time. Overall, all results show that adaptivity to available sonar data during a coverage mission and coverage replanning causes shorter coverage path/mission time in the assumed mission parameter space in which coverage area is longer than sonar swath, and not many interesting objects are expected at the seafloor.

Bibliography

[1] Blondel, P., The Handbook of Sidescan Sonar. Jointly published with Praxis Publishing, UK: Springer-Verlag Berlin Heidelberg, 2009.

[2] Suresh, B. N., Sivan, K., Navigation Guidance and Control System. New Delhi: Springer India, 2015, pages 581–661.

[3] Mandić, F., "Control of autonomous surface marine vessels for underwater vehicle localization using single range acoustic measurements", PhD thesis, 2019.

[4] Kapetanović, N., Đula Na đ, Mišković, N., Vukić, Z., "Towards Enhancing the Navigational Accuracy of UUVs Through Collaboration of Multiple Heterogeneous Marine Vehicles", in 2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV), 2018.

[5] Nad, D., "Guidance and control of autonomous underwater agents with acoustically aided navigation", PhD thesis, 2017.

[6] Kapetanović, N., "Qualifying examination: Informational gain guided model predictive motion planning for autonomous underwater vehicles", 2017.

[7] Kapetanović, N., Bibuli, M., Mišković, N., Caccia, M., "Real-time model predictive line following control for underactuated marine vehicles", IFAC-PapersOnLine, Vol. 50, No. 1, 2017, pages 12 374–12 379.

[8] Kapetanović, N., Miškovic, N., Na đ, D., Bibuli, M., Caccia, M., Bruzzone, G., "Development of model predictive guidance for underactuated marine vehicles: From simulations to experiments", in OCEANS 2017 - Aberdeen, June 2017, pages 1-6.

[9] Kapetanović, N., "MPC bazirano planiranje kretanja mobilnog vozila na neravnim terenima", Master's thesis, Elektrotehnički fakultet Sarajevo, Univerzitet u Sarajevu, 2015.

[10] Camacho, E. F., Bordons, C., Model Predictive Control. Springer-Verlag, London, 2004.

[11] Hoekstra, M., Vogelzang, M., Verbitskiy, E., Nijsten, M. W., "Health technology assessment review: Computerized glucose regulation in the intensive care unit - how to create artificial control", Critical Care, Vol. 13, No. 5, 2009, page 223.

[12] Bellman, R., Dynamic Programming, 1st ed. Princeton, NJ, USA: Princeton University Press, 1957.

[13] Kalman, R., "Contributions to the theory of optimal control", 1960.

[14] Qin, S. J., Badgwell, T. A., "An overview of industrial model predictive control technology", AIChE Symposium Series, Vol. 93, No. 316, 1997, pages 232–256.

[15] Allgöwer, F., Badgwell, T. A., Qin, J. S., Rawlings, J. B., Wright, S. J., "Nonlinear predictive control and moving horizon estimation—an introductory overview", in Advances in control. Springer London, 1999, pages 391–449.

[16] Bitmead, R. R., Gevers, M., Wertz, V., Adaptive optimal control : the thinking

man's GPC, ser. Prentice Hall international series in systems and control engineering. New York: Prentice Hall, 1990.

[17]Mayne, D. Q., Rawlings, J. B., Rao, C. V., Scokaert, P. O. M., "Survey constrained model predictive control: Stability and optimality", Automatica, Vol. 36, No. 6, Jun. 2000, pages 789–814.

[18]Bascetta, L., Cucci, D., Magnani, G., Matteucci, M., Osmankovic, D., Tahirovic, A., "Towards the implementation of a MPC-based planner on an autonomous all-terrain vehicle", Proceedings of Workshop on Robot Motion Planning: Online, Reactive, and in Real-time (IEEE/RJS IROS 2012), 2012, pages 1–7.

[19]Kapetanovic, N., Tahirovic, A., Magnani, G., "A fast cost-to-go map approximation algorithm on known large scale rough terrains", in Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, Sept 2015, pages 6279-6285.

[20]Tahirovic, A., Brkic, M., Magnani, G., Bascetta, L., "A planner for all terrain vehicles on unknown rough terrains based on the MPC paradigm and D*-like algorithm", in IEEE International Conference on Robotics and Automation ICRA 2014. Workshop on Modelling, Estimation, Perception and Control of All Terrain Mobile Robots. IEEE, June 2014.

[21]Miotto, P., Wilde, J., Menozzi, A., "UUV on-board path planning in a dynamic environment for the Manta test vehicle", in OCEANS 2003. Proceedings, Vol. 5, Sept 2003, pages 2454-2461 Vol.5.

[22]Caldwell, C. V., Dunlap, D. D., Collins, E. G., "Motion planning for an autonomous underwater vehicle via sampling based model predictive control", in OCEANS 2010 MTS/IEEE SEATTLE, Sept 2010, pages 1-6.

[23]Ono, M., Quadrelli, M., Huntsberger, T. L., "Safe maritime autonomous path planning in a high sea state", in 2014 American Control Conference, June 2014, pages 4727-4734.

[24]Huynh, V. T., Dunbabin, M., Smith, R. N., "Predictive motion planning for AUVs subject to strong time-varying currents and forecasting uncertainties", in 2015 IEEE International Conference on Robotics and Automation (ICRA), May 2015, pages 1144-1151.

[25]Hoy, M., Matveev, A. S., Savkin, A. V., "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey", Robotica, Vol. 33, No. 3, Mar 2015, page 463–497.

[26]Caldwell, C. V., Collins, E. G., Palanki, S., "Integrated guidance and control of AUVs using shrinking horizon model predictive control", in OCEANS 2006, Sept 2006, pages 1-6.

[27]Li, Z., Sun, J., Oh, S., "Path following for marine surface vessels with rudder and roll constraints: An MPC approach", in American Control Conference, June 2009, pages 3611-3616.

[28]Ghaemi, R., Oh, S., Sun, J., "Path following of a model ship using model predictive control with experimental verification", in Proceedings of the 2010 American Control Conference, June 2010, pages 5236-5241.

[29]Oh, S.-R., Sun, J., "Path following of underactuated marine surface vessels using line-of-sight based model predictive control", Ocean Engineering, Vol. 37, No. 2–3, 2010, pages 289 - 295.

[30]Liu, C., Sun, J., Zou, Z., "Integrated line of sight and model predictive control for

path following and roll motion control using rudder", Journal of Ship Research, Vol. 59, No. 2, 2015.

[31]Caccia, M., Bibuli, M., Bono, R., Bruzzone, G., "Basic navigation, guidance and control of an unmanned surface vehicle", Autonomous Robots, Vol. 25, No. 4, 2008, pages 349–365.

[32]Muske, K. R., Badgwell, T. A., "Disturbance modeling for offset-free linear model predictive control", Journal of Process Control, Vol. 12, No. 5, 2002, pages 617–632.

[33]Houska, B., Ferreau, H., Diehl, M., "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization", Optimal Control Applications and Methods, Vol. 32, No. 3, 2011, pages 298–312.

[34]Saggini, E., Zereik, E., Bibuli, M., Bruzzone, G., Caccia, M., Riccomagno, E., "Performance indices for evaluation and comparison of unmanned marine vehicles' guidance systems", IFAC Proceedings Volumes, Vol. 47, No. 3, 2014, pages 12 182–12 187.

[35]Plets, R., Dix, J., Bates, R., "Marine geophysics data acquisition, processing and interpretation", English Heritage, May 2013.

[36]Ludvigsen, M., Johnsen, G., Sørensen, A. J., Lågstad, P. A., Ødegård, Ø., "Scientific operations combining ROV and AUV in the Trondheim Fjord", Marine Technology Society Journal, Vol. 48, No. 2, 2014, pages 59-71.

[37]Choset, H., "Coverage for robotics – A survey of recent results", Annals of Mathematics and Artificial Intelligence, Vol. 31, No. 1, Oct 2001, pages 113–126.

[38]Galceran, E., Carreras, M., "A survey on coverage path planning for robotics", Robotics and Autonomous systems, Vol. 61, No. 12, 2013, pages 1258–1276.

[39]Kapetanović, N., Mišković, N., Tahirović, A., "Information gain-guided online coverage path planning for side-scan sonar survey missions", in 2018 26th Mediterranean Conference on Control and Automation (MED), June 2018, pages 1-9.

[40]Kapetanović, N., Mišković, N., Tahirović, A., Bibuli, M., Caccia, M., "A side-scan sonar data-driven coverage planning and tracking framework", Annual Reviews in Control, Vol. 46, 2018, pages 268 - 280.

[41]Kapetanović, N., Mišković, N., Tahirović, A., "Side-scan sonar data-driven coverage path planning: A comparison of approaches", in MTS/IEEE OCEANS '19 Marseille Conference and Exhibit, 2019, pages 1-6.

[42]Wang, Y., Lane, D. M., Falconer, G. J., "Two novel approaches for unmanned underwater vehicle path planning: constrained optimisation and semi-infinite constrained optimisation", Robotica, Vol. 18, No. 2, Mar 2000, page 123–142.

[43]Petillot, Y., Ruiz, I. T., Lane, D. M., "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar", IEEE Journal of Oceanic Engineering, Vol. 26, No. 2, Apr 2001, pages 240-251.

[44]Yilmaz, N. K., Evangelinos, C., Lermusiaux, P. F. J., Patrikalakis, N. M., "Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming", IEEE Journal of Oceanic Engineering, Vol. 33, No. 4, Oct 2008, pages 522-537.

[45]Antonelli, G., Chiaverini, S., Finotello, R., Schiavon, R., "Real-time path planning and obstacle avoidance for RAIS: an autonomous underwater vehicle", IEEE Journal of Oceanic Engineering, Vol. 26, No. 2, Apr 2001, pages 216-227.

[46] Hernández, J. D., Vidal, E., Vallicrosa, G., Galceran, E., Carreras, M., "Online path planning for autonomous underwater vehicles in unknown environments", in 2015 IEEE International Conference on Robotics and Automation (ICRA), May 2015, pages 1152-1157.

[47] Alvarez, A., Caiti, A., Onken, R., "Evolutionary path planning for autonomous underwater vehicles in a variable ocean", IEEE Journal of Oceanic Engineering, Vol. 29, No. 2, April 2004, pages 418-429.

[48] Aghababa, M. P., "3D path planning for underwater vehicles using five evolutionary optimization algorithms avoiding static and energetic obstacles", Applied Ocean Research, Vol. 38, 2012, pages 48 - 62.

[49] l. Zhang, G., m. Jia, H., "Global path planning of auv based on improved ant colony optimization algorithm", in 2012 IEEE International Conference on Automation and Logistics, Aug 2012, pages 606-610.

[50] Garau, B., Alvarez, A., Oliver, G., "Path planning of autonomous underwater vehicles in current fields with complex spatial variability: An A* approach", in Proceedings of the 2005 IEEE International Conference on Robotics and Automation, April 2005, pages 194-198.

[51] Petres, C., Pailhas, Y., Patron, P., Petillot, Y., Evans, J., Lane, D., "Path planning for autonomous underwater vehicles", IEEE Transactions on Robotics, Vol. 23, No. 2, April 2007, pages 331-341.

[52] Soulignac, M., "Feasible and optimal path planning in strong current fields", IEEE Transactions on Robotics, Vol. 27, No. 1, Feb 2011, pages 89-98.

[53] Zeng, Z., Lian, L., Sammut, K., He, F., Tang, Y., Lammas, A., "A survey on path planning for persistent autonomy of autonomous underwater vehicles", Ocean Engineering, Vol. 110, Part A, 2015, pages 303 - 313.

[54] Ousingsawat, J., Earl, M. G., "Modified lawn-mower search pattern for areas comprised of weighted regions", in 2007 American Control Conference, July 2007, pages 918-923.

[55] Tahirovic, A., Astolfi, A., "A convergent solution to the multi-vehicle coverage problem", in 2013 American Control Conference, June 2013, pages 4635-4641.

[56] Sadat, S. A., Wawerla, J., Vaughan, R. T., "Recursive non-uniform coverage of unknown terrains for UAVs", in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sept 2014, pages 1742-1747.

[57] Scott, K., Dai, R., Kumar, M., "Occlusion-aware coverage for efficient visual sensing in unmanned aerial vehicle networks", in 2016 IEEE Global Communications Conference (GLOBECOM), Dec 2016, pages 1-6.

[58] Araújo, J. F., Sujit, P. B., Sousa, J. B., "Multiple UAV area decomposition and coverage", in 2013 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), April 2013, pages 30-37.

[59] Tahirovic, A., Brkic, M., Bostan, A., Seferagic, B., "A receding horizon scheme for constrained multi-vehicle coverage problems", in 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Oct 2016.

[60] Hert, S., Tiwari, S., Lumelsky, V., "A terrain-covering algorithm for an AUV", Autonomous Robots, Vol. 3, No. 2, Jun 1996, pages 91–119.

[61] Jung, Y.-S., Lee, K.-W., Lee, S.-Y., Choi, M. H., Lee, B.-H., "An efficient underwater coverage method for multi-AUV with sea current disturbances", International Journal of Control, Automation and Systems, Vol. 7, No. 4, Aug 2009,

pages 615–629.

[62] Paull, L., SaeediGharahbolagh, S., Seto, M., Li, H., "Sensor driven online coverage planning for autonomous underwater vehicles", in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct 2012.

[63] Belbachir, A., Ingrand, F., Lacroix, S., "A cooperative architecture for target localization using multiple AUVs", Intelligent Service Robotics, Vol. 5, No. 2, Apr 2012, pages 119–132.

[64] Galceran, E., Carreras, M., "Coverage path planning for marine habitat mapping", in 2012 Oceans, Oct 2012, pages 1-8.

[65] Galceran, E., Nagappa, S., Carreras, M., Ridao, P., Palomer, A., "Uncertainty-driven survey path planning for bathymetric mapping", in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nov 2013, pages 6006-6012.

[66] Abreu, N., Cruz, N., Matos, A., "Accounting for uncertainty in search operations using AUVs", in 2017 IEEE Underwater Technology (UT), Feb 2017, pages 1-8.

[67] Yu, X., Roppel, T. A., Hung, J. Y., "An optimization approach for planning robotic field coverage", in IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society, Nov 2015, pages 004 032-004 039.

[68] Hoffman, R., Krotkov, E., "Terrain roughness measurement from elevation maps", in Proc. of SPIE, Vol. 1195, 1990, pages 104-114.

[69] Loh, J. E., Elkaim, G. H., Curry, R. E., "Roughness map for autonomous rovers", in Proc. of ACC, 2013.

[70] Du Preez, C., "A new arc–chord ratio (ACR) rugosity index for quantifying three-dimensional landscape structural complexity", Landscape Ecology, Vol. 30, No. 1, 2015, pages 181–192.

[71] Ahsan, N., Williams, S. B., Jakuba, M., Pizarro, O., Radford, B., "Predictive habitat models from AUV-based multibeam and optical imagery", in OCEANS 2010 MTS, Sept 2010, pages 1-10.

[72] Iagnemma, K., Dubowsky, S., Mobile Robots in Rough Terrain: Estimation, Motion Planning, and Control with Application to Planetary Rovers, 1st ed. Springer Publishing Company, Incorporated, 2010.

[73] Mandić, F., Mišković, N., Vukić, Z., "Range–only navigation – maximizing system observability by using extremum seeking", IFAC-PapersOnLine, Vol. 48, No. 16, 2015, pages 101 - 106, 10th IFAC Conference on Manoeuvring and Control of Marine Craft MCMC 2015.

[74] Bayat, M., Aguiar, A. P., "AUV range-only localization and mapping: Observer design and experimental results", in 2013 European Control Conference (ECC), July 2013, pages 4394-4399.

[75] H. Huang, W., "Optimal line-sweep-based decompositions for coverage algorithms", in Proceedings - IEEE International Conference on Robotics and Automation, Vol. 1, 02 2001, pages 27 - 32.

[76] Tahirovic, A., Magnani, G., Kuwata, Y., "An approximate of the cost-to-go map on rough terrains", in 2013 IEEE International Conference on Mechatronics, ICM 2013, 02 2013, pages 242-247.

[77] Chapple, P., "Automated detection and classification in high-resolution sonar imagery for autonomous underwater vehicle operations executive". Australian Government, Department of Defence, Defence Science and Technology Organi-

sation, Maritime Operations Division, 2008.

[78]Kapetanović, N., , Mišković, N., Tahirović, A., "Saliency and anomaly: Transition of concepts from natural images to side-scan sonar images", IFAC-PapersOnLine, 2020.

[79]Sharma, P., "Evaluating visual saliency algorithms: Past, present andfuture", Journal of Imaging Science and Technology, Vol. 59, No. 5, 2015, pages 50 501-1-50 501-17.

[80]Chandola, V., Banerjee, A., Kumar, V., "Anomaly detection: A survey", ACM Comput. Surv., Vol. 41, No. 3, Jul. 2009, pages 15:1–15:58.

[81]Ji, Y., Zhang, H., Wu, Q. J., "Salient object detection via multi-scale attention cnn", Neurocomputing, Vol. 322, 2018, pages 130 - 140.

[82]Huang, X., Shen, C., Boix, X., Zhao, Q., "Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks", in 2015 IEEE International Conference on Computer Vision (ICCV), Dec 2015, pages 262-270.

[83]Pan, J., Ferrer, C. C., McGuinness, K., O'Connor, N. E., Torres, J., Sayrol, E., i Nieto, X. G., "Salgan: Visual saliency prediction with generative adversarial networks", 2017.

[84]Reed, S., Petillot, Y., Bell, J., "An automatic approach to the detection and extraction of mine features in sidescan sonar", IEEE Journal of Oceanic Engineering, Vol. 28, No. 1, Jan 2003, pages 90-105.

[85]Mishne, G., Talmon, R., Cohen, I., "Graph-based supervised automatic target detection", IEEE Transactions on Geoscience and Remote Sensing, Vol. 53, No. 5, May 2015.

[86]Chandran, V., Elgar, S., Nguyen, A., "Detection of mines in acoustic images using higher order spectral features", Oceanic Engineering, IEEE Journal of, Vol. 27, 08 2002, pages 610 - 618.

[87]Zhu, B., Wang, X., Chu, Z., Yang, Y., Shi, J., "Active learning for recognition of shipwreck target in side-scan sonar image", Remote Sensing, Vol. 11, No. 3, 2019.

[88]Zheng, L., Tian, K., "Detection of small objects in sidescan sonar images based on pohmt and tsallis entropy", Signal Processing, Vol. 142, 07 2017.

[89]Goldman, A., Cohen, I., "Anomaly detection based on an iterative local statistics approach", in 2004 23rd IEEE Convention of Electrical and Electronics Engineers in Israel, Sep. 2004, pages 440-443.

[90]Noiboar, A., Cohen, I., "Anomaly detection based on wavelet domain garch random field modeling", IEEE Transactions on Geoscience and Remote Sensing, Vol. 45, No. 5, May 2007, pages 1361-1373.

[91]Acosta, G. G., Villar, S. A., "Accumulated ca–cfar process in 2-d for online object detection from sidescan sonar data", IEEE Journal of Oceanic Engineering, Vol. 40, No. 3, July 2015, pages 558-569.

[92]Rutledge, J., Yuan, W., Wu, J., Freed, S., Lewis, A., Wood, Z. J., Gambin, T., Clark, C. M., "Intelligent shipwreck search using autonomous underwater vehicles", 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pages 1-8.

[93]Zhu, P., Isaacs, J., Fu, B., Ferrari, S., "Deep learning feature extraction for target recognition and classification in underwater sonar images", in 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Dec 2017, pages 2724-2731.

[94]Einsidler, D., Dhanak, M., Beaujean, P.-P. J., "A deep learning approach to target

recognition in side-scan sonar imagery", OCEANS 2018 MTS/IEEE Charleston, 2018, pages 1-4.

[95]Kaeli, J. W., "Real-time anomaly detection in side-scan sonar imagery for adaptive auv missions", in 2016 IEEE/OES Autonomous Underwater Vehicles (AUV), Nov 2016, pages 85-89.

[96]Mishne, G., Cohen, I., "Multiscale anomaly detection using diffusion maps", IEEE Journal of Selected Topics in Signal Processing, Vol. 7, No. 1, Feb 2013, pages 111-123.

[97]Zhai, Y., Shah, M., "Visual attention detection in video sequences using spatiotemporal cues", in Proceedings of the 14th ACM International Conference on Multimedia, ser. MM '06. New York, NY, USA: ACM, 2006, pages 815–824.

[98]Harel, J., Koch, C., Perona, P., "Graph-based visual saliency", in Proceedings of the 19th International Conference on Neural Information Processing Systems, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pages 545–552.

[99]Itti, L., Koch, C., Niebur, E., "A model of saliency-based visual attention for rapid scene analysis", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 11, Nov 1998, pages 1254-1259.

[100]Harel, J., Simple Itti-Koch Saliency Method, 2019 (accessed February, 2019), available at:http://www.vision.caltech.edu/~harel/share/gbvs.php

[101]Hou, X., Harel, J., Koch, C., "Image signature: Highlighting sparse salient regions", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 34, No. 1, Jan 2012.

[102]Liu, Y., Chen, L., Wang, H., Jiang, L., Zhang, Y., Zhao, J., Wang, D., Zhao, Y., Song, Y., "An improved differential box-counting method to estimate fractal dimensions of gray-level images", Journal of Visual Communication and Image Representation, Vol. 25, No. 5, 2014, pages 1102 - 1111.

[103]Koenig, N., Howard, A., "Design and use paradigms for gazebo, an open-source multi-robot simulator", in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE), Vol. 3, Sep. 2004, pages 2149-2154 vol.3.

[104]Manhães, M. M. M., Scherer, S. A., Voss, M., Douat, L. R., Rauschenbach, T., "UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation", in OCEANS 2016 MTS/IEEE Monterey, Sep. 2016, pages 1-8.

[105]project, S., "Side-scan sonar simulator developed under the smarc project", available at:https://github.com/smarc-project/smarc_simulations

[106]Manhães, M. M. M., Scherer, S. A., Voss, M., Douat, L. R., Rauschenbach, T., "UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation", in OCEANS 2016 MTS/IEEE Monterey. IEEE, sep 2016, available at:https://doi.org/10.1109%2Foceans.2016.7761080

[107]Bruno, F., Lagudi, A., Collina, M., Medaglia, S., Kalamara, S., Kalamara, P., Kourkoumelis, D., Nad, D., Kapetanovic, N., Vasilijevic, A., Miskovic, N., "Optoacoustic 3D reconstruction and virtual diving on the Peristera shipwreck", in Dive in Blue Growth 2019, 2019, pages 1-11.

[108]Miskovic, N., Nad, D., Rendulic, I., "Tracking divers: An autonomous marine surface vehicle to increase diver safety", IEEE Robotics Automation Magazine, Vol. 22, No. 3, 2015, pages 72-84.

[109]Vasilijevic, A., Nad, D., Miskovic, N., "Autonomous Surface Vehicles as Position-

ing and Communications Satellites for the Marine Operational Environment — Step toward Internet of Underwater Things", in 2018 IEEE 8th International Conference on Underwater System Technology: Theory and Applications (USYS), 2018, pages 1-5.

[110]Lončar, I., Babić, A., Arbanas, B., Vasiljević, G., Petrović, T., Bogdan, S., Mišković, N., "A heterogeneous robotic swarm for long-term monitoring of marine environments", Applied Sciences, Vol. 9, Apr 2019, page 1388.

[111]Vasilijevic, A., Buxton, B., Sharvit, J., Stilinovic, N., Nad, D., Miskovic, N., Planer, D., Hale, J., Vukic, Z., "An ASV for coastal underwater archaeology: The Pladypos survey of Caesarea Maritima, Israel", in OCEANS 2015 - Genova, 2015, pages 1-7.

[112]Djapic, V., Nad, D., "Using collaborative autonomous vehicles in mine countermeasures", in OCEANS'10 IEEE SYDNEY, 2010, pages 1-7.

[113]Miskovic, N., Nad, D., Stilinovic, N., Vukic, Z., "Guidance and control of an overactuated autonomous surface platform for diver tracking", in 21st Mediterranean Conference on Control and Automation, 2013, pages 1280-1285.

[114]Goričanec, J., Kapetanović, N., Vatavuk, I., Hrabar, I., Kurtela, A., Anić, M., Vasilijević, G., Bolotin, J., Kožul, V., Stuhne, D., Cukon, M., Glavić, N., Bogdan, S., Orsag, M., Gledec, G., Petrović, T., Antolović, N., Kozina, B., Mišković, N., Kovačić, Z., "Heterogeneous autonomous robotic system in viticulture and mariculture – project overview", Proceedings of the 16th International Conference on Telecommunications - ConTEL, Zagreb, Croatia, 2021, pages 1-8.

[115]Kapetanović, N., Nad, D., Mišković, N., "Towards a heterogeneous robotic system for autonomous inspection in mariculture", Proceedings of the OCEANS 2021 Conference and Exposition, San Diego – Porto (Hybrid), 2021, pages 1-6.

[116]Bruzzone, G., Bibuli, M., Zereik, E., Ranieri, A., Caccia, M., "Cooperative adaptive guidance and control paradigm for marine robots in an emergency ship towing scenario", International Journal of Adaptive Control and Signal Processing, 2016.

[117]Feezor, M. D., Sorrell, F. Y., Blankinship, P. R., Bellingham, J. G., "Autonomous underwater vehicle homing/docking via electromagnetic guidance", IEEE Journal of Oceanic Engineering, Vol. 26, No. 4, Oct 2001, pages 515-521.

[118]Gode, T., "Long baseline ranging acoustic positioning system", Master's thesis, 2015.

[119]Font, E. G., Bonin-Font, F., Negre, P.-L., Massot, M., Oliver, G., "USBL integration and assessment in a multisensor navigation approach for AUVs", IFAC-PapersOnLine, Vol. 50, No. 1, 2017, pages 7905 - 7910, 20th IFAC World Congress.

[120]Bosch, J., Gracias, N., Ridao, P., Istenič, K., Ribas, D., "Close-range tracking of underwater vehicles using light beacons", Sensors, Vol. 16, No. 4, 2016.

[121]Fallon, M. F., Papadopoulos, G., Leonard, J. J., Patrikalakis, N. M., "Cooperative AUV navigation using a single maneuvering surface craft", The International Journal of Robotics Research, Vol. 29, No. 12, 2010, pages 1461-1474.

[122]Vasilijevic, A., Nad, D., Mandic, F., Miskovic, N., Vukic, Z., "Coordinated navigation of surface and underwater marine robotic vehicles for ocean sampling and environmental monitoring", IEEE/ASME Transactions on Mechatronics, Vol. 22, No. 3, June 2017, pages 1174-1184.

[123]Paull, L., Saeedi, S., Seto, M., Li, H., "AUV navigation and localization: A re-

view", IEEE Journal of Oceanic Engineering, Vol. 39, No. 1, Jan 2014, pages 131-149.

[124]Pinto, J., Calado, P., Braga, J., Dias, P., Martins, R., Marques, E., Sousa, J., "Implementation of a control architecture for networked vehicle systems", IFAC Proceedings Volumes, Vol. 45, No. 5, 2012, pages 100 - 105, 3rd IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles.

[125]Simon, D., Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & Sons, 2006.

[126]Einicke, G. A., "Smoothing, filtering and prediction: Estimating the past, present and future", 2012.

[127]Simon, D., Optimal state estimation: Kalman, H-infinity and nonlinear approaches. Wiley-Interscience, 2006.

[128]Đula Na đ, Mišković, N., Mandić, F., "Navigation, guidance and control of an overactuated marine surface vehicle", Annual Reviews in Control, Vol. 40, 2015, pages 172 - 181.

[129]Pastore, T., Vukić, Z., Na đ, D., Kapetanović, N., "A method for measuring navigation accuracy of submerged uuvs", 2019, pages 1-2.

[130]Klaic, Z., Rubinić, J., Kapelj, S., "Review of research on Plitvice Lakes, Croatia in the fields of meteorology, climatology, hydrology, hydrogeochemistry and physical limnology", Geofizika, Vol. 35, 12 2018, page 189.

[131]Franić, D., "Plitvička jezera i njihova okolica", author's own edition, 1910, pages 1-439.

[132]Golubić, S., Violante, C., Plenković-Moraj, A., Grgasović, T., "Travertines and calcareous tufa deposits: an insight into diagenesis", Geologia Croatica, Vol. 61, No. 2, 2008, pages 363-378.

[133]Polšak, A., Šparica, M., Crnko, J., M., J., "Osnovna geološka karta SFRJ 1:100.000, Tumač za list Bihać (Basic Geological Map of SFRY 1:100000, Bihac sheet) l33–116", Institut za geološka istraživanja Zagreb, Savezni geološki zavod, 1978, page 52.

[134]Gavazzi, A., "Prilozi za limnologiju plitvica, prirodoslovna istraživanja hrvatske i slavonije", Jugoslavenska akademija znanosti i umjetnosti, Vol. 14, 1919, page 3–37.

[135]Petrik, M., "Prinosi hidrologiji plitvica, u nacionakni park plitvička jezera", Poljoprivredni nakladni zavod Zagreb, 1958, page 49–173.

[136]Kapetanović, N., Vasilijević, A., Zubčić, K., "Assessing the current state of a shipwreck using an autonomous marine robot: Szent istvan case study", in Proceedings of the 17th International Conference on Distributed Computing and Artificial Intelligence (DCAI) 2020, 2020, pages 1-10.

[137]Pacheco-Ruiz, R., Adams, J., Pedrotti, F., Grant, M., Holmlund, J., Bailey, C., "Deep sea archaeological survey in the black sea – robotic documentation of 2,500 years of human seafaring", Deep Sea Research Part I: Oceanographic Research Papers, Vol. 152, 2019, page 103087.

[138]Drap, P., Merad, D., Seinturier, J., Mahiddine, A., Peloso, D., Boï, J.-M., Chemisky, B., Long, L., Garrabou, J., "Underwater programmetry for archaeology and marine biology: 40 years of experience in Marseille, France", in Proceedings of the DigitalHeritage 2013 - Federating the 19th Int'l VSMM, 10th Eurographics GCH, and 2nd UNESCO Memory of the World Conferences, Plus Special Ses-

sions fromCAA, Arqueologica 2.0 et al., Vol. 1, 2013, pages 97-104.

[139] Santagati, C., Inzerillo, L., Di Paola, F., "Image-based modeling techniques for architectural heritage 3D digitalization: Limits and potentialities", in International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, Vol. 40, 2013, pages 555-560.

[140] Gracias, N., Ridao, P., Garcia, R., Escartín, J., L'Hour, M., Cibecchini, F., Campos, R., Carreras, M., Ribas, D., Palomeras, N., Magi, L., Palomer, A., Nicosevici, T., Prados, R., Hegedüs, R., Neumann, L., de Filippo, F., Mallios, A., "Mapping the Moon: Using a lightweight AUV to survey the site of the 17th century ship La Lune", in 2013 MTS/IEEE OCEANS - Bergen, 2013, pages 1-8.

[141] Roman, C., Inglis, G., Rutter, J., "Application of structured light imaging for high resolution mapping of underwater archaeological sites", in OCEANS'10 IEEE Sydney, OCEANSSYD 2010, 2010.

[142] Pribičević, B., Medak, D., Kordić, B., "Application of two-frequencies bathymetry in determination of travertine layer", Geodetski list, Vol. 61(85), No. 1, 2007, pages 1-18.

[143] Pribičević, B., Medak, D., Đapo, A., "Integration of modern geodetic and hydrographic measurement methods in karst areas of croatia", Ekscentar : časopis studenata Geodetskog fakulteta Sveučilišta u Zagrebu, Vol. 12, 2010, pages 58-63.

[144] Kordić, B., Lužar-Oberiter, B., Matoš, B., Miculinić, K., Markić, N., Željko, R., "Application of geodetic techniques in geological studies of the Plitvice Lakes National Park", in 6th Croatian Geological Congress, 2019.

[145] Kordić, B., Lužar-Oberiter, B., Pikelj, K., Matoš, B., Vlastelica, G., "Integration of Terrestrial Laser Scanning and UAS Photogrammetry in Geological studies: Examples from Croatia", Periodica Polytechnica Civil Engineering, Vol. 63, No. 4, 2019, pages 989-1003.

[146] Kapetanović, N., Kordić, B., Vasilijević, A., Na đ, D., Mišković, N., "Autonomous vehicles mapping plitvice lakes national park, croatia", Remote Sensing, Vol. 12, No. 22, 2020, page 20.

[147] McCarthy, J., Benjamin, J., Winton, T., van Duivenvoorde, W., 3D Recording and Interpretation for Maritime Archaeology. Springer International Publishing, 04 2019.

[148] Bruno, F., Miskovic, N., Nad, D., Kapetanovic, N., Lagudi, A., Aiello, R., Lupia, M., Cario, G., "New technologies for improving the diver experience in underwater cultural sites", in Proceedings of the International Conference on Archaeology and Tourism Sense and sustainability, 2019, pages 1-6.

[149] Kapetanović, N., Vasilijević, A., Na đ, D., Zubčić, K., Mišković, N., "Marine robots mapping the present and the past: Unraveling the secrets of the deep", Remote Sensing, Vol. 12, No. 23, 2020, page 39.

[150] Vasilijević, A., Kapetanović, N., Mišković, N., "Inspection of submerged structures", in Proceedings of the 17th IMEKO TC 10 and EUROLAB Virtual Conference, 2020, pages 1-5.

[151] Pizer, S. M., Johnston, R. E., Ericksen, J. P., Yankaskas, B. C., Muller, K. E., "Contrast-limited adaptive histogram equalization: speed and effectiveness", in [1990] Proceedings of the First Conference on Visualization in Biomedical Computing, May 1990, pages 337-345.

[152] Mandić, D., Milanković, G., Klarić, A., A Protected Subaqueous Site S.M.S. Szent

Istvan the Austro-Hungarian Tegettoff Class Battleship. Povijesni muzej Istre-Pula, 2001.

[153]Hummel, R., "Image enhancement by histogram transformation", Computer Graphics and Image Processing, Vol. 6, No. 2, 1977, pages 184 - 195.

Acronyms

ACADO Toolkit for Automatic Control and Dynamic Optimization 15,22

AGV Autonomous Ground Vehuicle 28

ARM Advanced Reduced Instruction Set Computing Machine 159

ASV Autonomous Surface Vehicle 29

AUV Autonomous Underwater Vehicle 5,6,12,23, 24,27,29,70, 71,74,78,99, 159,164

BA-CPP Basic Accordion Coverage Path Planning 25,46,48–58, 60,61,63,66, 67,69,185, 191

CL-CPP Classical static overlap-all-sonar-swaths coverage maneuver 25,30–32,37, 39–47,49– 61,63–69, 184–186

CLAHE Contrast-limited adaptive histogram equalization algorithm 161

CNR Consiglio Nazionale delle Ricerche 104,105

CoG centre of gravity 116

CPP Coverage Path Planning 24–27,29– 32,34–37, 40,45,46, 56,57,60, 65–69,71, 184,186

CPU Central processing unit 43,67,77, 159,162,163

CSG Constructive Solid Geometry 26

CSI Camera Serial Interface 162

CUDA Compute Unified Device Architecture 77

DC direct current 162

DoF degree of freedom 5,117

DP Direct path 114

DUNE Unified Navigation Environment 99,111,119

Nomenclature

Term	Description
C	coverage area
F	harmonic mean metric
L	lawnmower manouver lane length
W	lawnmower manouver lane width
α_{lm}	ratio between lawnmower lane length and width
β	heading error in line following
$\boldsymbol{F}$	linearized system model matrix
$\boldsymbol{H}$	linearized measurement model matrix
$\boldsymbol{K}$	Kalman filter gain matrix
$\boldsymbol{P}$	estimation error covariance matrix
$\boldsymbol{Q}$	process noise covariance matrix
$\boldsymbol{Q}$	measurement noise covariance matrix
$\boldsymbol{f}$	vector of nonlinear system functions
$\boldsymbol{h}$	vector of nonlinear measurement functions
$\boldsymbol{u}$	generalized vector of system inputs
$\boldsymbol{v}$	generalized vector of measurement noise
$\boldsymbol{w}$	generalized vector of process noise
$\boldsymbol{x}$	generalized vector of system states
$\boldsymbol{y}$	generalized vector of system outputs
$\mathcal{N}_{i,j}$	set of 8-connected neighborhood patches of the patch $p_i j$ including the patch $p_{i,j}$ itself
$\mathcal{T}$	sea floor terrain (backscatter intensity) map
d_{int}	integral of the cross-track line following error state
d	cross-track line following error
$e_{CL-CPP}^{best\,WRbNF-CPP}$	analytical upper bound relative improvement of the generated coverage path by the algorithm named in superscript w.r.t. the coverage path length generated by the algorithm named in the subscript
$e_{CL-CPP}^{worst\,WRbNF-CPP}$	analytical lower bound relative improvement of the generated coverage path by the algorithm named in superscript w.r.t. the coverage path length generated by the algorithm named in the subscript
$e_{CL-CPP}^{WRbNF-CPP}$	relative improvement of the generated coverage path by the algorithm named in superscript w.r.t. the coverage path length generated by the algorithm named in the subscript
p_{better}	percentage of simulations in which the analysed coverage algorithm generated shorter coverage paths than the CL-CPP algorithm
$p_{i,j}$	cost map patch index

Term	Description
$p_{obj}^{critical}$	percentage of awnmower lanes in the coverage area containing interesting objects for which the analysed coverage path planning algorithm has the same performance as the non-adaptive CL-CPP algorithm
p_{obj}	percentage of lawnmower lanes in the coverage area containing interesting objects
$r_{i,j}$	roughness of a cost map patch $p_{i,j}$
$s_{i,j}$	smoothness of a cost map patch $p_{i,j}$
w_{sss}	side-scan sonar horizontal range

List of Figures

List of Tables

Biography

Nadir Kapetanović was born on 11th of November 1990 in Sarajevo, Bosnia and Herzegovina. He graduated from the University of Sarajevo Faculty of Electrical Engineering in 2015, at the Department of Automation and Electronics under the mentorship of assoc. prof. Adnan Tahirović. The title of his Master's thesis was "MPC-based motion planning for a mobile robot on rough terrains".

In 2015. he entered the Doctoral studies at the University of Zagreb Faculty of Electrical Engineering and Computing at the Department of Automation and Computer Engineering under the mentorship of prof. Nikola Mišković and assoc. prof. Adnan Tahirović. He is an employee of FER since the beginning of 2015. starting as a junior researcher in the Laboratory for Underwater Systems and Technologies. Since 2017, he's been involved in several research projects, namely MORUS (2015-2019), EXCELLABUST (2016-2018), BLUEMED (2016-2019), e-URready4OS (2017-2018), CUV-ME (2017 and 2021), EUMarineRobots (2018-2021), HEKTOR (2020-2022), and INNOVAMARE (2020-2022).

Nadir was a guest researcher at the Consiglio Nazionale delle Ricerche in Genoa, Italy in the scope of EXCELLABUST project. Since 2018 he's a member of IEEE and OES societies, and was involved in the founding of the IEEE OES UNIZG Student Branch Chapter where in which he's the Secretary. In the scope of HEKTOR project (2020-2023) he is actively involved not only in the research, but also in technical project management together with the coordinator prof. Zdenko Kovačić and prof. Nikola Mišković. At the 13th international innovation fair AGRO ARCA held in 2022 in Croatia he and his team were awarded with Grand Prix for the innovation named „Autonomous catamaran for mariculture applications". The same innovation received the prize of the World Intellectual Property Organization as the best international innovation at ISIF 2022 International Innovation Fair.

His research interests include model predictive control, path and coverage planning of autonomous marine vehicles, bathymetric models of the seabed, fotogramemtric 3D models of objects, state estimation methods, as well as signal and image processing. He authored 5 journal articles and 20 conference papers, and he also participated in reviewing dozens of research papers.

List of publications (Nadir Kapetanović)

Journal papers

[1]Akram, W., Casavola, A., Kapetanović, N., Miškovic, N. (2022) A Visual Servoing Scheme for Autonomous Aquaculture Net Pens Inspection Using ROV. Sensors, 22 (9), 1-23.

[2]Kapetanović, N., Goričanec, J., Vatavuk, I., Hrabar, I., Stuhne, D., Vasiljević, G., Kovačić, Z., Mišković, N., Antolović, N., Anić, M., Kozina, B. (2022) Heterogeneous Autonomous Robotic System in Viticulture and Mariculture: Vehicles Development and Systems Integration. Sensors, 22 (8), 1-29.

[3]Kapetanović, N., Vasilijević, A., Na đ, Đ., Zubčić, K., Mišković, N. (2020) Marine Robots Mapping the Present and the Past: Unraveling the Secrets of the Deep. Remote Sensing, 12 (23), 3902, 39.

[4]Kapetanović, N., Kordić, B., Vasilijević, A., Na đ, Đ., Mišković, N. (2020) Autonomous Vehicles Mapping Plitvice Lakes National Park, Croatia. Remote Sensing, 12 (22), 3683, 20.

[5]Kapetanović, N., Mišković, N., Tahirović, A., Bibuli, M., Caccia, M. (2018). "A side-scan sonar data-driven coverage planning and tracking framework". Annual Reviews in Control, 46, 268-280.

Conference papers

[1]Fabijanić, M., Kapetanović, N. Mišković, N. (2022) Biofouling Estimation in Mariculture. U: Proceedings of the OCEANS 2022 Hampton Roads Conference and Exposition.

[2]Kapetanović, N., Oreč, M., Mišković, N., Fausto Ferreira (2022) Landing Platform for Autonomous Inspection Missions in Mariculture Using an ASV and a UAV. U: Proceedings of the 14th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles.

[3]Kapetanović, N., Krčmar, K., Mišković, N., Na đ, Đ. (2022) Tether Management System for Autonomous Inspection Missions in Mariculture Using an ASV and an ROV. U: Proceedings of the 14th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles.

[4]Kapetanović, N., Na đ, Đ., Lončar, I., Slošič, V., Mišković, N. (2022) Acoustical Underwater Localization of a Remotely Operated Vehicle in Mariculture. U: Proceedings of the 17th International Conference on Intelligent Autonomous Systems (IAS-17).

[5]Kapetanović, N., Na đ, Đ., Mišković, N. (2021) Towards a Heterogeneous Robotic System for Autonomous Inspection in Mariculture. U: Proceedings of the OCEANS 2021 Conference and Exposition.

[6]Borković, G., Fabijanić, M., Magdalenić, M., Malobabić, A., Vuković, J., Zieliński, I., Kapetanović, N., Kvasić, I., Babić, A., Mišković, N. (2021) Underwater ROV software for fish cage inspection. U: Proceedings of the 44th International ICT Convention MIPRO.

[7]Goričanec, J., Kapetanović, N., Vatavuk, I., Hrabar, I., Vasiljević, G., Gledec, G., Stuhne, D., Bogdan, S., Orsag, M., Petrović, T. (2021) Heterogeneous autonomous

robotic system in viticulture and mariculture – project overview. U: Antonić, M., Babić, J. (ur.)Proceedings of the 16th International Conference on Telecommunications: ConTEL 2021

[8]Vasilijević, A., Kapetanović, N., Mišković, N. (2020) Inspection of Submerged Structures. U: Proceedings of the 17th IMEKO TC 10 and EUROLAB Virtual Conference.

[9]Kapetanović, N., Vasilijević, A., Zubčić, K. (2020) Assessing the current state of a shipwreck using an autonomous marine robot: Szent Istvan case study. U: Proceedings of the 17th International Conference on Distributed Computing and Artificial Intelligence (DCAI) 2020.

[10]Kapetanović, N., Mišković, N., Tahirović A. (2020). "Saliency and Anomaly: Transition of Concepts from Natural Images to Side-Scan Sonar Images". IFAC World Congress. Berlin, Germany, 1-6.

[11]Bruno, F., Lagudi, A., Collina, M., Medaglia, S., Kalamara, S., Kalamara, P., Kourkoumelis D., Nađ, Đ., Kapetanović, N., Vasilijević, A., Mišković, N. (2019). "Opto-acoustic 3D Reconstruction and Virtual Diving on the Peristera Shipwreck". Dive in Blue Growth 2019 Proceedings. Athens, Greece, p. 1-11.

[12]Kapetanović, N., Mišković, N., Tahirović, A., Kvasić, I. (2019). "Side-Scan Sonar Data-Driven Coverage Path Planning: A Comparison of Approaches". MTS/IEEE OCEANS 2019. Marseille, France, 1-6.

[13]Bruno, F., Mišković, N., Na đ, Đ., Kapetanović, N., Lagudi A., Aiello, R., Lupia, M., Cario, G. (2019). "New technologies for improving the diver experience in underwater cultural sites". Proceedings of the International Conference on Archaeology and Tourism Sense and sustainability, ICAT 2019. Zagreb, Croatia. p. 1-6

[14]Pastore T., Vukić, Z., Na đ, Đ., Kapetanović, N. (2019). "A method for measuring navigation accuracy of submerged UUVs ". Proceedings of the International Conference on Undersea Defence Technology, UDT 2019. Stockholm, Sweden.

[15]Kapetanović, N., Na đ, Đ., Mišković, N., Vukić, Z. (2018). "Towards Enhancing the Navigational Accuracy of UUVs Through Collaboration of Multiple Heterogeneous Marine Vehicles". 2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV). Porto, Portugal, 1-6.

[16]Kapetanović, N., Mišković, N., Tahirović, A. (2018). "Information Gain-Guided Online Coverage Path Planning for Side-Scan Sonar Survey Missions". 26th Mediterranean Conference on Control and Automation (MED). Zadar, Croatia, 1-9.

[17]Kapetanović, N., Bibuli, M., Mišković, N., Caccia, M. (2017). "Real-time model predictive line following control for underactuated marine vehicles". 2017, IFAC (International Federation of Automatic Control Conference). Toulouse, France, 1-6

[18]Kapetanović, N., Mišković, N., Na đ, D., Bibuli, M., Caccia, M., Bruzzone, G. (2017). "Development of model predictive guidance for underactuated marine vehicles: From simulations to experiments". MTS/IEEE OCEANS2017. Aberdeen, UK, 1-6.

[19]Kapetanović, N., Tahirović, A., Magnani, G. (2015). "A fast cost-to-go map approximation algorithm on known large scale rough terrains". 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Hamburg, Germany, 6279-6285.

[20]Kapetanović, N., Osmić, N., Konjicija, S. (2014). "Optimization of membership functions of Sugeno-Takagi fuzzy logic controllers with two inputs and one output

using genetic algorithms". 2014 X International Symposium on Telecommunications (BIHTEL). Sarajevo, Bosnia and Herzegovina, 1-7.

Životopis

Nadir Kapetanović rođen je 11. studenog 1990. godine u Sarajevu. Diplomirao je 2015. godine na Elektrotehničkom fakultetu Univerziteta u Sarajevu, smjer Automatika i elektronika pod mentorstvom izv. prof. dr. sc. Adnana Tahirovića s diplomskim radom pod nazivom „MPC bazirano planiranje kretanja mobilnog vozila na neravnim terenima".

U ak. god. 2015/2016 upisuje doktorski studij na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu na Zavodu za automatiku i računalno inženjerstvo pod mentorstvom prof. dr. sc. Nikole Miškovića i izv. prof. dr. sc. Adnana Tahirovića. Početkom 2017. godine zapošljava se na FER-u kao mladi istraživač u Laboratoriju za podvodne sustave i tehnologije (LAPOST). Od 2015. godine do danas bio je uključen u rad na mnogim projektima: MORUS (2015-2019), EXCELLABUST (2016-2018), BLUEMED (2016-2019), e-URready4OS (2017-2018), CUV-ME (2017 i 2021), EU-MarineRobots (2018-2021), HEKTOR (2020-2022), i INNOVAMARE (2020-2022).

Godine 2016. bio je gostujući istraživač na Consiglio Nazionale delle Ricerche u Genovi, Italija u sklopu EXCELLABUST projekta. Od 2018. godine član je društava IEEE i OES, te je sudjelovao u osnivanju studentskog odjela IEEE Oceanic Engineering Society Sveučilišta u Zagrebu u kojem je na poziciji tajnika od osnutka. Na projektu HEKTOR (2020-2023) aktivno je uključen u istraživanje, ali i u vođenje projekta skupa sa koordinatorom prof. dr. sc. Zdenkom Kovačićem i prof. dr. sc. Nikolom Miškovićem. Na 13. međunarodnom sajmu inovacija AGRO ARCA 2022 dobitnik je glavne nagrade „Grand Prix" za inovaciju „Autonomni katamaran za marikulturu". Ista inovacija osvojila je nagradu svjetske organizacije za intelektualno vlasništvo (WIPO) za najbolju međunarodnu inovaciju sajma inovacija ISIF 2022.

Njegovi istraživački interesi uključuju modelski prediktivno upravljanje, planiranje putanje i pokrivenosti (ne samo) podvodnim morskim vozilima, batimetrijske modele morskog dna, fotogrametrijske 3D modele objekata na kopnu i podmorju, tehnike estimacije, kao i obradu signala i slike . Zajedno s kolegama iz LABUST-a koautor je 5 radova u znanstvenim časopisima i 20 radova na međunarodnim konferencijama, te je recenzirao desetke konferencijskih i žurnalskih članaka.